17

Sommario

RUBRICHE

NOVITA'

Direttore: Alessandro de Simone
Redazione/collaboratori: Giovanni Bellù, Andrea e Alberto Boriani, Giancarlo Castagna, Eugenio Coppari, Marco De Martino, Giancarlo Mariani, Enrico Scelsa, Fabio Sorgato, Danilo Toma
Segreteria di redazione: Maura Ceccaroli, Piera Perin
Impaginazione/Illustrazioni: Francesco Amatori, Renato Caruso
Composizioni: Systems Editoriale S.r.l.
Fotolito: Systems Editoriale S.r.l.
Direzione, redazione: V.le Famagosta, 75 - 20142 Milano - Tel. 02/8467348
Pubblicità: Milano: Mirco Croce (coordinatore), Michela Prandini, Giorgio Ruffoni, Claudio Tidone, Villa Claudio - Segretaria: Liliana De Giorgi
● Roma: Spazio Nuovo - via P. Foscari 70 - 00139 Roma - Tel. 06/8209679
Prezzo e abbonamento: prezzo per copia L. 3.000.
Abbonamento annuo (11 fascicoli) L. 28.000. Abbonamento annuo cumulativo alle riviste Computer e Commodore Computer Club L. 55.000.
I versamenti vanno indirizzati a: Systems Editoriale Srl mediante assegno bancario o utilizzando il c/c postale n. 31532203
Stampa: La Litografica S.r.l. - Busto Arsizio
Registrazione: Tribunale di Milano n. 370 del 2/1/82 - Direttore Responsabile: Michele Di Pisa
Sped. in abb. post. gr. III - Pubblicità inferiore al 70%
Distribuzione: MePe, via G. Carcano 32 - Milano

Da questo mese la rivista C.C.C. è disponibile anche su nastro

AVVERTENZE:

Numerosi lettori pongono domande molto simili tra loro. Gli interessati, pertanto, sono pregati di individuare le risposte ai loro quesiti tra le tante pubblicate che, per il motivo esposto, sono prive del nome dei richiedenti.

Colori indesiderati con le routine di Toma.

☐ **Il programma "SOLIDI IN ALTA RISOLUZIONE" (n. 14 di C.C.C.) una volta lanciato presenta lo sfondo non costituito da un sottile reticolato, ma da strisce verticali verdi e rosse larghe quanto un carattere. Come mai?**

● La colpa non è del programma bensì dell'uscita video del Commodore 64. Quando due puntini (pixel) di diverso colore sono contigui provocano il fastidioso effetto che lamenti.

Ciò è dovuto al fatto che il Commodore 64 non è un computer "professionale", ma un home computer dalle qualità inevitabilmente più modeste di un computer grafico. Un elaboratore che non presentasse l'inconveniente accennato costerebbe molto di più e richiederebbe, tra l'altro, l'uso di un monitor ad alta definizione

Per limitare il difetto senza rinunciare all'output grafico in Hi-Res, osserva i grafici in bianco e nero, routando al minimo la manopola del colore del TV.

Caratteri grafici poco leggibili

☐ **Perché nei vostri listati non usate un codice per i caratteri grafici come fanno le altre riviste?**

● Come puoi notare, da un pò di tempo pubblichiamo su ogni numero della rivista una intera pagina che non solo indica i caratteri grafici, ma anche il modo di ottenerli. Commodore Computer Club, infatti, preferisce... "invogliare" il lettore ad interpretare correttamente i programmi pubblicati perché, credetemi, anche in questo modo si impara a programmare: prestando la massima attenzione alle pubblicazioni.

Del resto, hai notato che per i lettori... pigri ci stiamo organizzando? Non solo abbiamo lanciato sul mercato una rivista su cassettta (Commodore Club) ma, addirittura, i programmi pubblicati sulla rivista che stai leggendo...

Azimut, chi era costui

☐ **Mi hanno detto che sul registratore c'è un azimut da regolare nel caso non si riesca a caricare un programma. Che cosa è?**

● L'azimut è l'altezza cui è necessario posizionare la testina di lettura/scrittura allo scopo di allinearla correttamente con la pista magnetica del nastro. Nei numeri scorsi, comunque, abbiamo più volte dato indicazioni su come comportarsi in questi casi.

Utility in basic

☐ **Come realizzare la funzione di renumerazione di linea e la fusione tra due programmi Basic?**

● Sul N. 15 di C.C.C. sono state pubblicate ben sette routine di semplice digitazione che qui riassumo:

- Renumber. Consente di renumerare nell'ordine desiderato qualsiasi programma Basic.
- Append. Consente di "fondere" due programmi l'uno di seguito all'altro.
- Find. Effettua una ricerca, all'interno di un programma, ed evidenzia le linee Basic in cui sono presenti particolari istruzioni, comandi, variabili eccetera.
- Dump. Dopo aver fatto girare un programma qualsiasi, elenca le variabili fino a quel momento interessate dall'elaborazione. Utile per rintracciare eventuali errori di programmatori (debug).
- Delete. Cancella le ultime linee di un programma Basic.
- GOSUB calcolato. Consente di utilizzare una sintassi del tipo: GOSUB (A*(B/(C+D)).
- GOSUB LABEL. Come (f) consente espressione simile a: GOSUB PRIMA ROUTINE e simili.

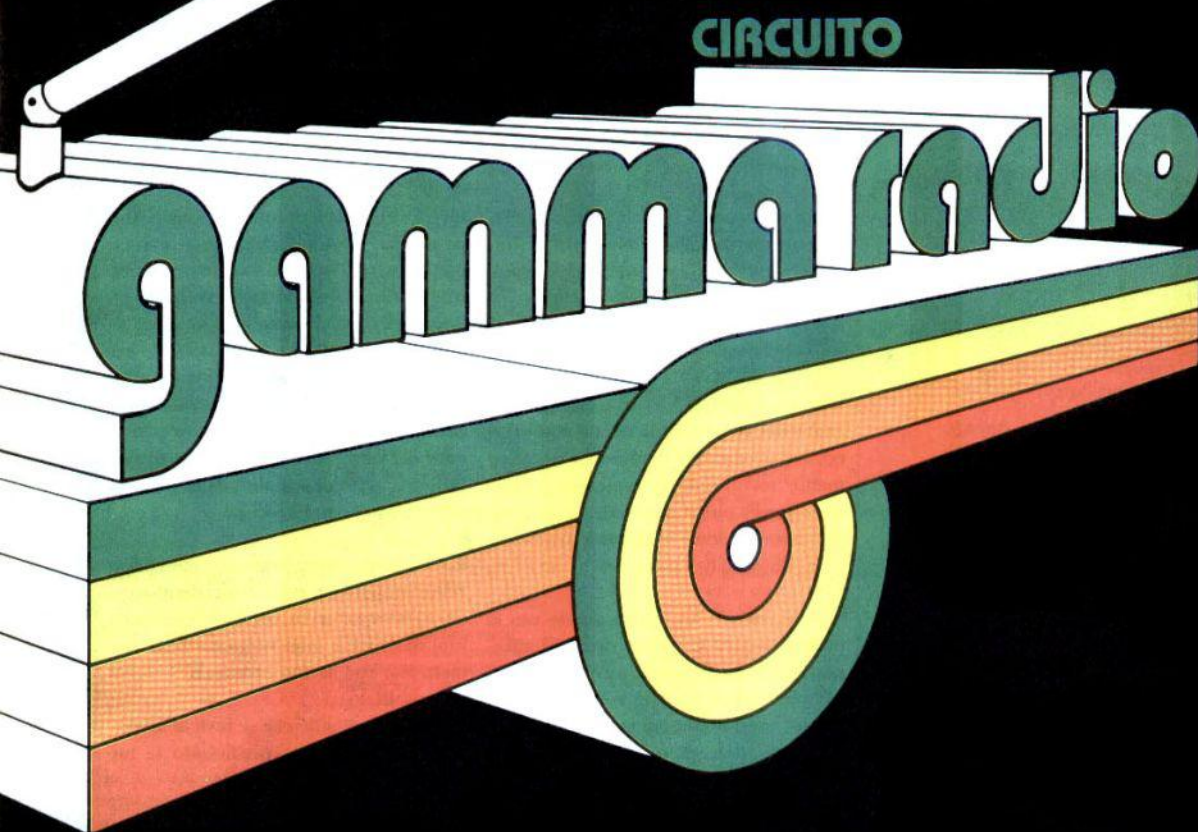
64 impazzito

☐ **A volte il mio Commodore 64 sembra impazzire e ciò avviene senza una ragione plausibile. A che cosa può esser dovuto un simile comportamento? (segue elenco di malfunzionamenti).**

● Rispondiamo elencando quelle che, a nostro parere, possono essere le cause di anomali comportamenti:

- Saldature "fredde". Alcuni esemplari (computer, drive, stampanti eccetera) contengono al loro interno componenti elettronici saldati al circuito stampato in modo non perfetto.

Quando si accende il computer tali elementi sono freddi ed il contatto elettrico risulta, per puro caso, assicurato. Dopo un po' di tempo, però, il calcolatore si riscalda e dilatazioni termiche delle saldature possono generare distacchi accidentali tra gli elementi mal fissati con

Segue a pag. 75

CONCESSIONARIA PER LA PUBBLICITÀ DI MILANO

RADIANT S.P.A.

CONCESSIONARIA PER LA PUBBLICITÀ DEL CIRCUITO

gamma italia S.P.A.

PALAZZO CANOVA CENTRO DIREZIONALE MILANO 2 - 20090 SEGRATE (MI)
TEL. 02/2155714 - 2155726 - 2155734

LOMBARDIA		*LIGURIA*		*PIEMONTE/VAL D'AOSTA*		*TOSCANA*	
Milano	95.9-92.8-97.1	Genova	96.25	Alessandria	104.3	Firenze	97.6-104.4
Bergamo	99.3	La Spezia	98.7	Cuneo	90.6-97.6	Livorno	98.2-97.3-100.6
Brescia	92-92.7			Novara	97.1	Massa C.	98.7
Como	97.1	*EMILIA ROMAGNA*				Pistoia	97.6-104.4
Cremona	99.3					Pisa	97.3
Pavia	95.9-97.1	Bologna	88.7	Aosta	91.8-92	Lucca	97.3
Varese	101.1	Modena	87.75				
		Parma	87.75			*LAZIO*	
		Piacenza	97.1				
		Reggio E.	87.75			Roma	99.5

Cio' che la gente vuole

Un regnante di qualche tempo fa soleva dire che, per governare senza problemi, era sufficiente accontentare il popolo con tre "effe": Festa, Farina e Forca.

Era infatti convinto, e purtroppo gli eventi gli davano ragione, che regalando, di tanto in tanto, banchetti (innaffiati, magari, da pessimo vino) o consegnando gratis del pane (meglio se in tempi di eccessiva produzione di grano) o semplicemente torturando e impiccando delinquenti nella pubblica piazza, i sudditi apprezzassero simili iniziative e trascurassero problemi ben più gravi.

Che cosa c'entra tutto questo con la nostra rivista? E' presto detto. Si affermava, tempo fa, che l'utente medio di personal computer non voleva altro che giochi, meglio se tutti eguali, meglio se capaci di attanagliare il giocatore al joystick per molte ore al giorno. Convinti, invece, che l'utente medio è di tipo "umano" e sicuri che durante la giornata svolge attività "pesanti" abbiamo, tempo addietro, proposto timidamente argomenti un po' diversi, di didattica, matematica, linguaggio macchina ed altre cose del genere. Il successo di tale iniziativa fu immediato e richieste di continuare sulla strada intrapresa continua senza sosta. Il continuo aumento della tiratura può, a onor del vero, esser dovuto anche ad altri fattori a volte di semplice determinazione, a volte di natura oscura.

Una cosa è certa: nel futuro pubblicheremo soltanto ciò che i lettori desiderano e tra i lettori ci sei tu, con le tue esigenze, i tuoi desideri, e (non vergognarti ad ammetterlo) con le tue incertezze e dubbi non risolti.

E stai tranquillo che se invii la scheda dell'ultima pagina, specificando le tue esigenze, verrai accontentato.

Alessandro de Simone

Un'iniziativa condotta con la nota rivista Computer

PROGRAMMO IN BASIC

Il linguaggio del futuro in un manuale rapido e completo di Clizio Merli
pagg. 224 (L. 9.000)

Il Basic, attualmente il linguaggio più conosciuto - adatto all'utilizzo su qualunque tipo di macchina e in particolare sul personal e gli home-computer - può essere appreso in poche ore con l'ausilio di questo agile manuale.

COME SCEGLIERE UN COMPUTER

Guida pratica per l'acquisto di un mini o di un micro computer professionale di Michele Di Pisa
pagg. 160 (L. 6.000)

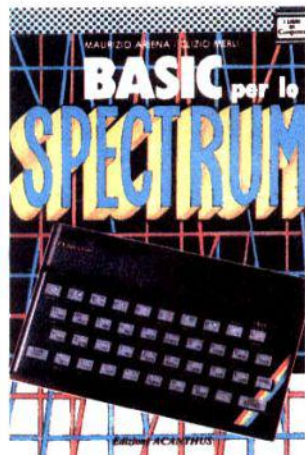
Quale modello scegliere tra gli oltre 600 computer commercializzati in Italia? La conoscenza delle caratteristiche delle varie macchine è indispensabile. Con un approccio a "menu" l'Autore vuol essere guida proprio in questa fase.

UTILITY E ROUTINE PER IL COMMODORE 64

di Gloriano Rossi
pagg. 192 (L. 9.000)

L'esecuzione di una istruzione BASIC può richiedere diverse centinaia di passi di programmi in linguaggio macchina. La dimensione dei programmi è ciò che intimidisce maggiormente l'utilizzatore medio di Commodore: aiutato da questo testo chiunque potrà affrontare senza problemi il processo di scrittura di un programma.

BASIC PER LO SPECTRUM

di Maurizio Ariena e Clizio Merli
pagg. 192 (L. 9.000)

Un libro per quanti hanno acquistato il computer ZX Spectrum della Sinclair e intendono sfruttarne appieno tutte le capacità, dall'hardware alla programmazione in assembly (linguaggio macchina).

I volumi, che sono comunque in vendita nelle migliori librerie di tutta Italia, possono anche essere richiesti direttamente all'Editore.
Importante: l'ordine minimo dovrà essere di L. 15.000.

DIZIONARIO DELL'INFORMATICA

Vocabolario Inglese-Italiano di Cultrera, Di Pisa, Giacomelli
pagg. 388 (L. 25.000)

Uno strumento indispensabile per chi si avvicina al mondo dell'informatica e per gli specialisti che hanno l'esigenza di accedere alla dinamica letteratura anglosassone.

Edizioni ACANTHUS

VIALE GRAN SASSO, 23 - 20131 MILANO

Inviatemi i seguenti volumi:

Titolo	quantità	prezzo unitario
spese postali		L. 2.000
	totale	L.

Pagherò contrassegno il dovuto (più L. 2.000 per contributo spese postali) al ricevimento. Potrò restituire i libri entro 8 giorni se non saranno di mio gradimento e avere il rimborso immediato.

COGNOME
NOME
VIA N.
C.A.P. CITTÀ
FIRMA
DATA

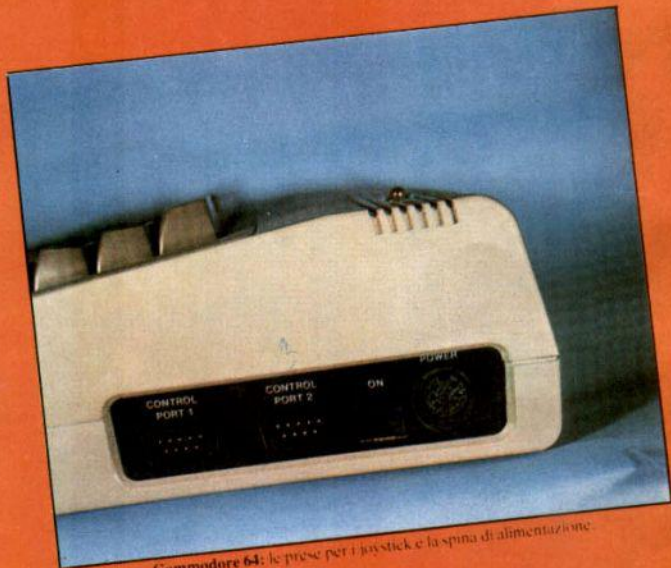
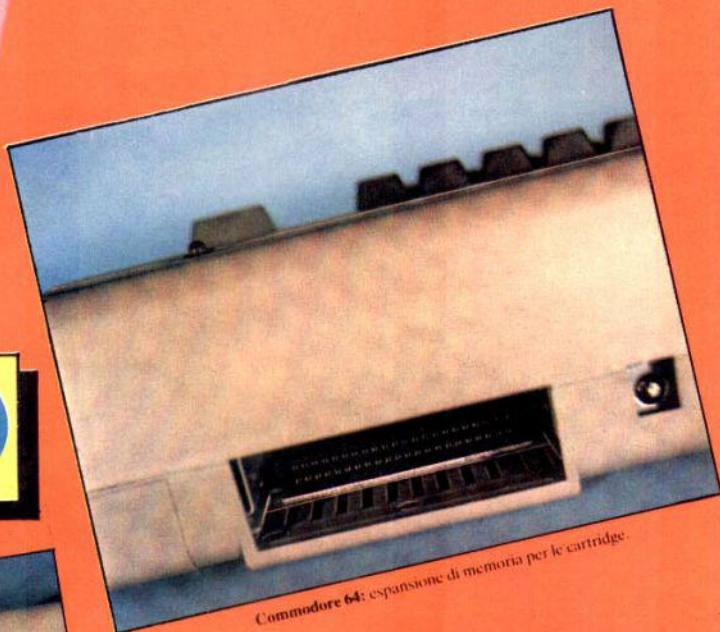
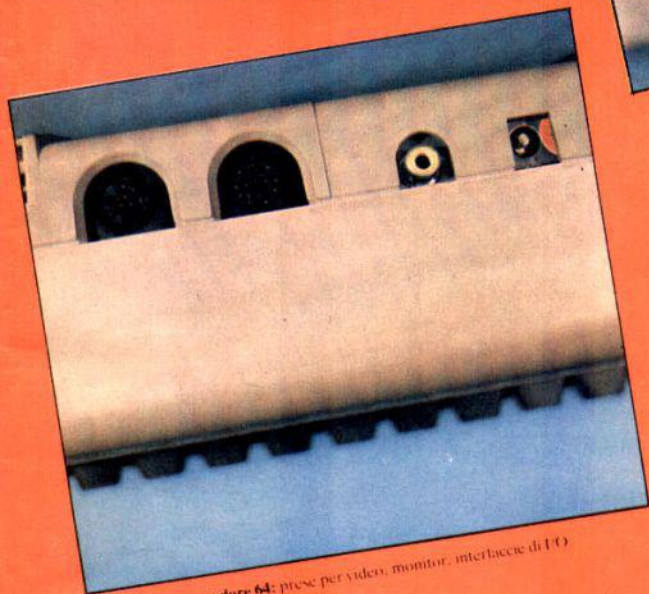
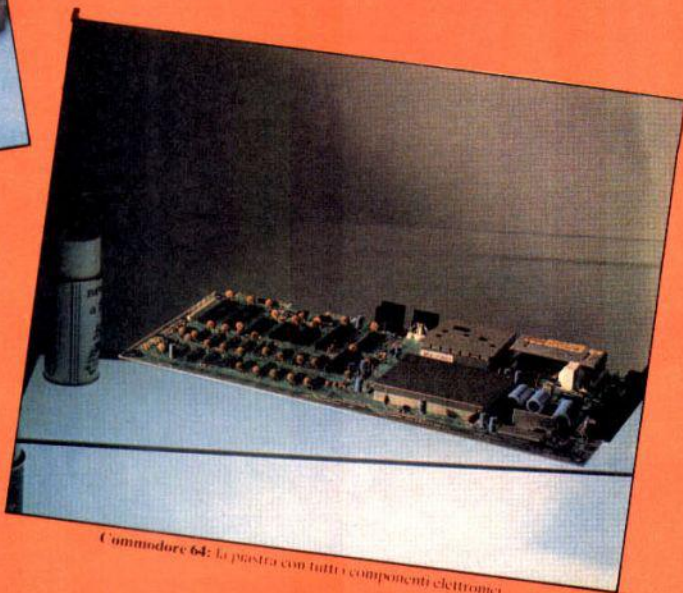
Scrivere in stampatello e spedire in busta chiusa.

LE IMMA

DI QUESTO FASC

Commodore 64: album di famiglia

Lo usiamo tutti i giorni. Commodore Computer Club ne parla tutti i mesi. Lo abbiamo sezionato, analizzato, smontato e rimontato mille volte. Ma lui, il Commodore 64, rimane sempre al centro delle nostre attenzioni. E ogni volta è come scoprirlo di nuovo. Ecco perchè in questo numero abbiamo deciso di proporre alcune 'foto tessera' del Commodore 64: frammenti, segnali, pose della nostra macchina ritratta da diverse angolazioni dall'obiettivo del fotografo. Dall'esterno e dall'interno. Dall'alto e di profilo.

Ma chi è il "padre" del Commodore 64? Le risposte sono diverse a seconda che si parli della sua architettura interna o della sua "carrozzeria". Nel primo caso, l'evoluzione è diretta. Dal Pet 2001 (il primo personal della storia), passando per la serie 3000 e 4000, fino al bivio: da una parte Vic 20 e Commodore 64, dall'altra la serie di livello superiore, la 8000. E allora, al primo padre è possibile dare un nome: Jack Paddle, il progettista del Pet 2001. Ma per quanto riguarda la "carrozzeria", lo styling... Al massimo si può dire che deriva dal lavoro di un team di progettisti. E, se proprio si vuole, si possono scoprire somiglianze con le prime macchine. Con buona pace di Darwin (ma esisterà, poi il darwinismo aziendale?) e del buon lavoro dei progettisti di casa Commodore.

Commodore 64: le prese per i joystick e la spina di alimentazione.

I programmi più interessanti spesso sono molto lunghi, un listato pubblicato è faticoso da leggere...

Commodore Computer Club vi offre un'alternativa: le cassette con tutti i programmi pubblicati sulla rivista.

Ogni nastro contiene il software di un numero di Commodore Computer Club a un prezzo incredibilmente basso: solo 5.800 lire (+ 1.000 lire per spese di spedizione).

Riceverete le cassette direttamente a casa vostra, utilizzando il coupon qui a fianco.

systems

Desidero ricevere le cassette con il software pubblicato sui seguenti numeri di Commodore Computer Club:

....

importo	L. ....
spese di spedizione	L. 1.000
Totale	L. ....

☐ ho versato l'importo sul c/c postale n. 30426209 (allego fotocopia della ricevuta di versamento)

☐ accludo assegno non trasf. n. .... (banca ....) intestato a Systems Editoriale, v.le Famagosta 75, 20142 MILANO

nome ....

cognome ....

via ....

CAP/città ....

Ritagliare e spedire in busta chiusa a: Systems Editoriale v.le Famagosta 75, 20142 Milano.

Abbonatevi a Commodore Computer Club

"KILL WOLF WITH SWORD"

di **Giuliano Boschi**

La lingua non è un ostacolo insormontabile negli adventures: i vocaboli usati sono in numero limitato e di facile apprendimento.

Vediamo quali sono i maggiori problemi che possiamo incontrare nello svolgimento di un adventure.

I verbi

Sono l'ostacolo maggiore: non è possibile dare delle regole precise, in quanto ogni programma può adoperare dei verbi particolari, al posto dei sinonimi più usati. Per esempio, in *The golden baton,* per tagliare dei rami, non si usa il solito CUT, bensì CHOMP, che letteralmente significa 'fare legna'.

L'unico consiglio è quello di consultare, nel miglior modo possibile, il vocabolario italiano-inglese e provare tutte le possibili traduzioni del verbo che vogliamo usare per realizzare una certa azione. Alla fine di questo articolo, troverete un piccolo vocabolario nel quale sono inseriti i verbi più comunemente usati negli adventures.

Particolare attenzione meritano i termini che vengono usati per trasferirsi da un luogo ad un altro: in genere si usano le iniziali della direzione geografica (**E**ast, **N**orth, **W**est, **S**outh, ed eventualmente le direzioni intermedie, NE, SO, etc.), verso la quale vogliamo dirigere, o UP e DOWN per su e giù. A volte, invece, bisogna digitare ENTER, GO, CLIMB, seguiti dal nome del posto in cui si vuole andare (es. ENTER HOME, GO CAVERN, CLIMB ROPE).

Alcuni capi di vestiario vanno indossati in particolari occasioni. In *Arrow of death (part 2),* bisogna indossare un'uniforme per non essere uccisi da una guardia; in *Message from Andromeda,* solo portando dei guanti possiamo raccogliere dei funghi, altrimenti pericolosamente mortali.

Molto usato è anche il verbo WAIT, aspettare, come ad esempio, in *The hobbit.*

Cose e persone vanno sempre esaminate (es. EXAMINE TABLE). Troveremo così altri oggetti, o accessi a luoghi che altrimenti non avremmo mai scoperto.

Frasi complesse

Non tutti i programmi sono costruiti con la stessa struttura logica: in alcuni potrebbe essere sufficiente digitare KILL WOLF, avendo già pre-

cedentemente trovato una spada, per poter uccidere un lupo.

In adventures più sofisticati e ormai sono molti, bisogna invece essere più precisi. Dovremo digitare, per esempio, KILL WOLF WITH SWORD (uccido il lupo con la spada), per ottenere il risultato desiderato. Oltre l'azione, dobbiamo quindi specificare con quale oggetto la realizziamo. Il sostantivo sarà così preceduto da una locuzione. Le più usate sono: con (WITH), a (AT, TO), in (IN, INTO) e attraverso (ACROSS).

Uso degli oggetti

E' necessario prestare molta attenzione alle cose che vengono trovate nel corso dell'adventure. Un oggetto che eventualmente non serve a nulla viene definito dagli inglesi un'*aringa rossa* (alcune si trovano, per esempio, nelle avventure della *Interceptor Micro's Software*).

In genere, comunque, tutti gli oggetti che possono essere raccolti devono poi venire usati; il problema è quello di riuscire a capire come farlo.

Se stiamo visitando un mondo di magia, potrebbe essere necessario strofinare (RUB) qualcosa (per esempio l'ORB di *Arrow of death† part 1*).

Alcuni oggetti si formano assemblandone insieme altri: una freccia è composta da un'asta, una punta e una piuma.

Se una porta è chiusa, può essere necessario sbloccare la serratura con l'apposita chiave (UNLOCK), prima di poterla aprire.

Gli esempi potrebbero continuare all'infinito: è allora opportuno ricordare che la logica o la fantasia, a seconda dei casi, devono aiutarci nei momenti difficili.

Un altro suggerimento generale: digitiamo una o più parole e il computer ci risponde di non averne capito il senso, vuol dire che conosce quelle parole, ma noi non le abbiamo usate nel modo giusto. Se invece risponde di non conoscerle, dovremo indirizzarci verso altre soluzioni.

Comandi speciali

Si tratta di sostantivi o verbi che se digitati, pur non influenzando direttamente l'esito dell'adventure, consentono di utilizzare opzioni indispensabili in questo tipo di giochi. I più comuni sono i seguenti (non necessariamente presenti in tutte le produzioni):

INVENTORY permette di sapere quali oggetti stiamo portando con noi e quali, eventualmente, stiamo indossando;

SCORE se nel gioco sono previsti punteggi, ci dice a quale livello siamo arrivati (per esempio 150 punti su 1000, oppure dà una percentuale);

LOOK ci ridescrive il luogo in cui ci troviamo; se nell'adventure vi è una parte grafica, questa viene di nuovo disegnata;

QUIT permette di ricominciare il gioco;

SAVE permette di salvare la situazione per poter continuare l'adventure in seguito, dal punto in cui è stata interrotta;

LOAD carica una situazione precedentemente salvata;

HELP fornisce eventuali suggerimenti per risolvere situazioni particolarmente difficili: abusandone si finisce spesso a non ottenere più niente e, a volte, a venir dileggiati dal computer.

Per concludere si può dire che negli adventure assume una importanza determinante l'esperienza: più si provano, più si giocano e più apprendiamo tecniche e nozioni per risolverli.

Nel prossimo numero vedremo metodi diversi per compilare le mappe del mondo che stiamo esplorando, fase indispensabile per poter correttamente portare a termine le missioni che ci vengono, di volta in volta, assegnate dal tipo di gioco.

Ricordiamo che per qualsiasi chiarimento o domanda sugli adventure potete scrivere direttamente all'autore: Giuliano Boschi , via F. Massi 12, 00152 Roma.

Elenco dei verbi più usati negli adventures

abbassare	*lower*
accendere	*light*
adattare	*fit*
andare	*go*
aprire	*open*
arrampicarsi	*climb*
aspettare	*wait*
attraversare	*through*
bere	*drink*
caricare	*load*
chiudere	*close*
colpire	*shoot*
correre	*run*
dare	*gift, offer*
dire	*say*
entrare	*enter*
esaminare	*examine*
girare	*turn*
guardare	*look*
indossare	*wear*
lanciare	*throw*
legare	*tie*
leggere	*read*
levarsi	*remove*
mangiare	*eat*
muovere	*move*
nuotare	*swim*
offrire	*gift, offer*
pregare	*pray*
prendere	*take, get*
riempire	*fill*
rompere	*break, shoot*
saltare	*jump*
sbloccare	*unlock*
scavare	*dig*
scrivere	*write*
soffiare	*blow*
slegare	*untie*
sparare	*fire*
spegnere	*extinguish*
spingere	*push*
strofinare	*rub*
tagliare	*cut*
tirare	*pull*
toccare	*touch*
uccidere	*kill*
usare	*use*
uscire	*out*
volare	*fly*

COME OTTENERE I CARATTERI SPECIALI DEL COMPUTER COMMODORE

Codice	Tasti	Codice	Tasti
= [NERO]	CTRL + 1 BLK	= [ARANC]	C= + 1 BLK
= [BIANCO]	CTRL + 2 WHT	= [MARR]	C= + 2 WHT
= [ROSSO]	CTRL + 3 RED	= [ROSA]	C= + 3 RED
= [AZZUR]	CTRL + 4 CYN	= [GRIGIO1]	C= + 4 CYN
= [VIOLA]	CTRL + 5 PUR	= [GRIGIO2]	C= + 5 PUR
= [VERDE]	CTRL + 6 GRN	= [VERDE]	C= + 6 GRN
= [BLU]	CTRL + 7 BLU	= [CELESTE]	C= + 7 BLU
= [GIALLO]	CTRL + 8 YEL	= [GRIGIO3]	C= + 8 YEL
= [TF2]	f2	= [TF1]	f1
= [TF4]	f4	= [TF3]	f3
= [TF6]	f6	= [TF5]	f5
= [TF2]	f8	= [TF7]	f7
= [DOWN]	CRSR	= [UP]	SHIFT + CRSR
= [RVS]	CTRL + 9 RVS ON	= [RVOFF]	CTRL + 0 RVS OFF
= [HOME]	CLR HOME	= [CLEAR]	SHIFT + CLR HOME
= [RIGHT]	CRSR	= [LEFT]	SHIFT + CRSR
= [DEL]	INST DEL	π = [PI]	SHIFT + ↑ π

IN MOLTI LISTATI PUBBLICATI IN QUESTA RIVISTA FIGURANO ALCUNI CARATTERI "SPECIALI" INDICHIAMO, SU QUESTA PAGINA, IL MODO DI OTTENERLI

PIEMONTE
Aba Elettronica
Via Fossati, 5/c
10141 Torino
Tel. 011/332065

Gruppo Sistemi Torino S.r.l.
Strada Torino, 90 h
10092 Beinasco (TO)
Tel. 011/651114

LOMBARDIA
Sirius Technology
Via Imperia, 23
20142 Milano
Tel. 02/8467304

Eledra System S.p.A.
Via Ferruccio, 2
20100 Milano
Tel. 02/3492010

VENETO-VENEZIA GIULIA
Corel Italiana S.r.l.
Via Mercatovecchio, 28
33100 Udine
Tel. 0432/291466-480857

Cash S.r.l.
Via Noventa Vicentina, 2
36100 Vicenza
Tel. 0444/507155

Vecomp S.r.l.
Via Chioda, 76
37136 Verona
Tel. 045/583711

Seda s.a.s.
Via Sighele, 7/l
38100 Trento
Tel. 0461/984564

LIGURIA
Siragusa Giuseppe
Via Milano, 85/a
16126 Genova
Tel. 010/261655

EMILIA ROMAGNA
S.H.R. S.r.l.
Casella Postale 275
48100 Ravenna
Tel. 0544/463200

Tempo Reale
Via Centotrecento, 1/A
40126 Bologna
Tel. 051/270701

Maser s.a.s.
Via Corticella, 177
40128 Bologna
Tel. 051/326420

TOSCANA
It-Lab S.r.l.
Via XXIV Maggio, 101
56100 Pisa
Tel. 050/501359

M.T.S. s.a.s.
V.le Guidoni, 93/Z
50100 Firenze
Tel. 055/410996

E.V.M.
Via Marconi, 9/A
52025 Montevarchi (AR)
Tel. 0575/982513

Indirizzi sicuri al ser del tuo Co

Oggi trovi in tutta Italia una grande rete di centri di assistenza tecnica Commodore. Sono gli unici centri autorizzati per assistere i computer Commodore, sia i sistemi che gli home computer con le relative periferiche, e vi operano tecnici competenti e preparati. Questi centri sono perciò in grado di garantirti un'assistenza tempestiva, totalmente affidabile, e un servizio molto efficiente e professionale. Tutto ciò ti permetterà di ottenere sempre il massimo

sistenza
mmodore

*e professionali
vizio
mmodore...*

di prestazioni e di sicurezza nell'uso del tuo computer Commodore.
Il tuo Commodore è costruito per non darti mai problemi; può però capitare di aver bisogno di un intervento tecnico; in tutti i casi, per essere maggiormente garantito, rivolgiti esclusivamente ai Centri di assistenza elencati qui.

MARCHE
I.M.P. Computers S.r.l.
Via Conti, 1
60100 Ancona
Tel. 071/804227-8

UMBRIA
Studio System s.a.s.
Via D'Andreotto, 49
06100 Perugia
Tel. 075/754964-753353

LAZIO
Atlas System S.r.l.
Via Marconi, 17
01100 Viterbo
Tel. 0761/224688

Kiber Italia S.r.l.
P.le Asia, 21
00144 Roma EUR
Tel. 06/5916438-5929590

Discom S.n.c.
Via della Pineta Sacchetti, 165
00168 Roma
Tel. 06/6279132

Computer Service Italia S.r.l.
Via Baldassarre Orero, 50
00195 Roma
Tel. 06/4382252

ABRUZZO
Pragma System S.r.l.
Via Tiburtina, 57
64100 Pescara
Tel. 085/5088301

CAMPANIA
Gamma Electronics S.r.l.
Via Naz. delle Puglie, Km 36,266
80013 Casalnuovo (NA)
Tel. 081/8421927

Computer Market s.a.s.
Parco S. Paolo Is. 9
80100 Napoli
Tel. 081/76722222

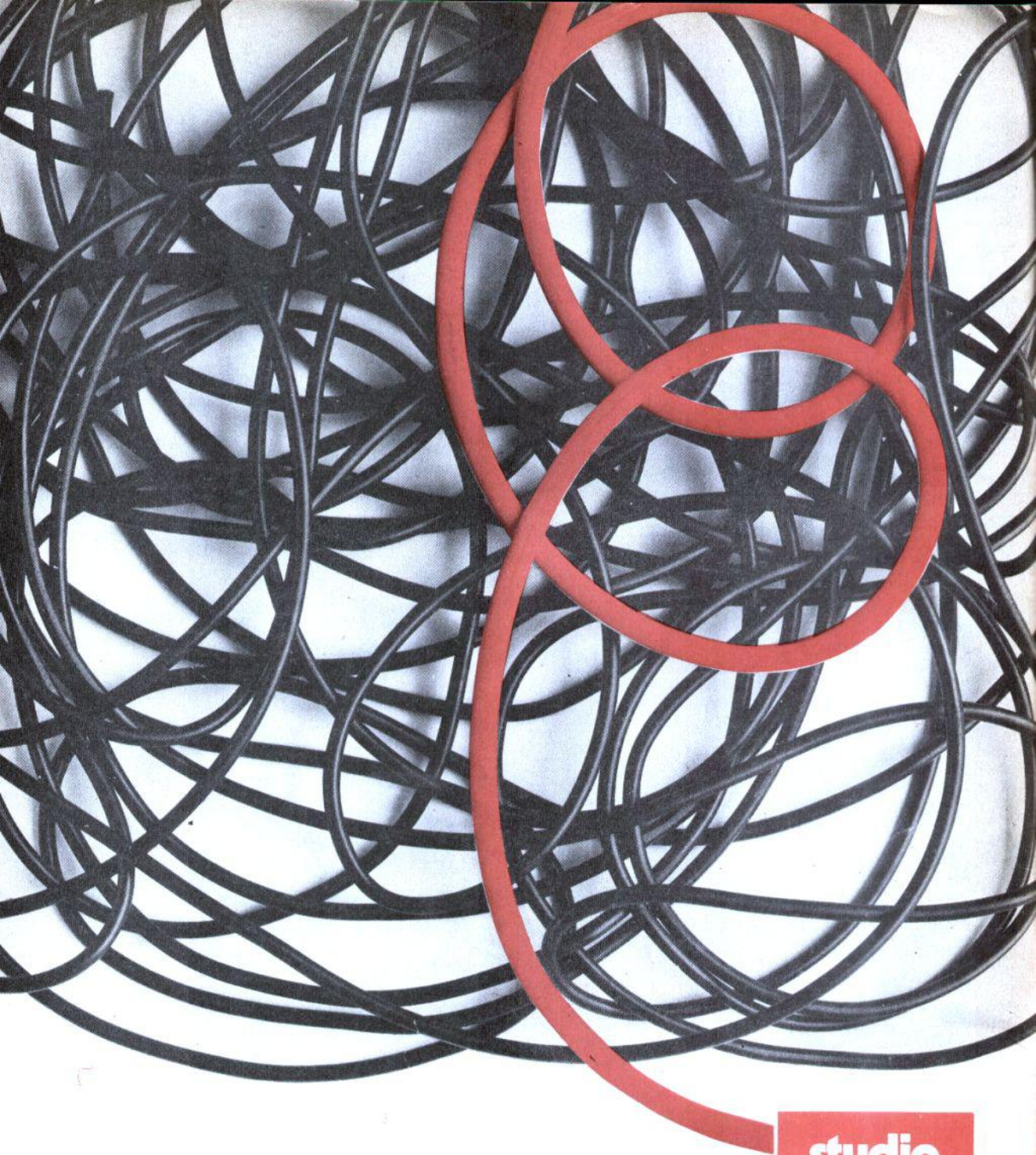
CALABRIA
Sirangelo Computers S.r.l.
Via Parisio, 25
87100 Cosenza
Tel. 0984/75741

Control System s.a.s.
Via S. Francesco da Paola, 49/D
89100 Reggio Calabria
Tel. 0965/94466

SICILIA
Edilcomput Progetti
Via La Farina, 141 Is. L
98100 Messina
Tel. 090/2928268

Elettronica Delta
Via Messina, 413/B
95100 Catania
Tel. 095/373946-370170

C.H.P. S.r.l.
Via Mondini, 3
90143 Palermo
Tel. 091/260780

SARDEGNA
S.I.I.
Via S. Lucifero, 95
09100 Cagliari
Tel. 070/663746

BLITZ

Scopo del gioco: distruggere i palazzi sganciando le bombe con l'aereo prima che esso stesso vada a schiantarsi contro di essi.

Ostacoli: palazzi, numerosi all'inizio del gioco. Il gioco termina se distruggete i palazzi o se vi sbattete contro.

Andrea Campione

```
1 REM*****************
2 REM*    BLITZ       *
3 REM*DI ANDREA       *
4 REM*  CAMPIONE      *
7 REM*PER VIC 20      *
8 REM*****************
9 POKE36869,240
10 POKE36879,98
11 POKE36878,15
12 PRINT"███====VIC====BLITZ======"
13 PRINT"█████GAME CREATED BY"
14 PRINT"ANDREA CAMPIONE"
15 PRINT"████PRESS F1 FOR"
16 PRINT"INSTRUCTION"
17 GETA$:IFA$<>"█"THENGOTO17
18 PRINT"█DISTRUGGI I PALAZZI DI"
19 PRINT"NEW YORK LANCIANDO LE"
20 PRINT"BOMBE CON UN QUALUNQUE"
21 PRINT"TASTO!"
22 PRINT"███F1 TO BEGINN........"
23 PRINT"████(C)BY SOFTAC 1984"
24 GETA$:IFA$<>"█"THENGOTO24
25 READP
26 IFP=-1THENGOTO40
27 READD
28 POKE36875,P
29 FORN=1TOD:NEXT
30 POKE36875,0
31 FORN=1TO20:NEXT
32 GOTO25
40 Q=7702:P=0
41 POKE36869,255
42 FORI=7168TO8185:POKEI,0:NEXT
43 FORI=7176TO7183:READA:POKEI,A:NEXT
44 FORI=7184TO7191:READB:POKEI,B:NEXT
45 FORI=7192TO7199:READC:POKEI,C:NEXT
46 PRINT" A A A A A A A A A A"
47 PRINT" A A A A A A A A A A"
48 PRINT" A A A A A A A AAA A"
49 PRINT" A A A A A A A AAA A"
50 PRINT" AAA AAA AAA A AAA A"
51 PRINT" AAA AAA AAA A AAA A"
52 PRINT" AAA AAA AAA AAAAA A"
53 PRINT" AAA AAA AAA AAAAA A"
54 PRINT" AAAAAAA AAAAAAAAA A"
55 PRINT"███1UP="P
56 GOSUB100
57 IFQ+1>8185THENGOTO2000
58 IFPEEK(Q+1)=1THENGOTO1000
59 GOSUB100
60 POKE36874,128
61 POKEQ,0:POKEQ+1,2:Q=Q+1
62 FORX=1TO100:NEXT
63 POKE36874,0
```

```
64 GOTO56
100 IFPEEK(197)=32THENGOTO200
101 IFPEEK(197)=64THENRETURN
200 W=Q+22
201 IFW+22>8163THENPOKEW,0:RETURN
202 IFPEEK(W+22)=1THENGOTO300
203 IFQ+1>8185THENGOTO2000
204 IFPEEK(Q+1)=1THENGOTO1000
205 POKE36874,128
206 POKE36876,250
207 POKEQ,0:POKEQ+1,2:Q=Q+1
208 POKEW,0:POKEW+22,3:W=W+22
209 FORX=1TO100:NEXT
210 POKE36874,0
211 POKE36876,0
212 GOTO201
300 H=INT(10*RND(1))
301 FORZ=1TOH
302 IFW+22>8185THENRETURN
303 P=P+100:PRINT"██1UP="P
304 POKEW,0:POKEW+22,0:W=W+22
305 POKE36877,200
306 FORX=1TO20:NEXT
307 POKE36877,0
308 NEXTZ
309 RETURN
1000 POKE36879,110
1001 POKE36877,200
1002 FORX=1TO1000:NEXT
1003 POKE36877,0
1004 PRINT"███PECCATO!
     HAI COZZATO!!!"
1005 POKE36877,0
1006 GOTO3000
2000 FORX=1TO1000:NEXT
2001 PRINT"███CONGRATULAZIONI!!!!!"
2002 GOTO3000
3000 PRINT"██F1 TO BEGINN....."
3001 GETA$:IFA$<>"■"THENGOTO3001
3002 RUN
10000 DATA235,100,235,100,232,100
10001 DATA235,100,127,100,228,80,
      127,100,228,80
10002 DATA235,100,240,100,239,100
10003 DATA235,100,127,500
10004 DATA-1
20000 DATA255,255,145,145,255,145,
      145,255
20001 DATA255,16,23,255,127,63,0,0
20002 DATA102,102,24,60,126,60,24,24

READY.
```

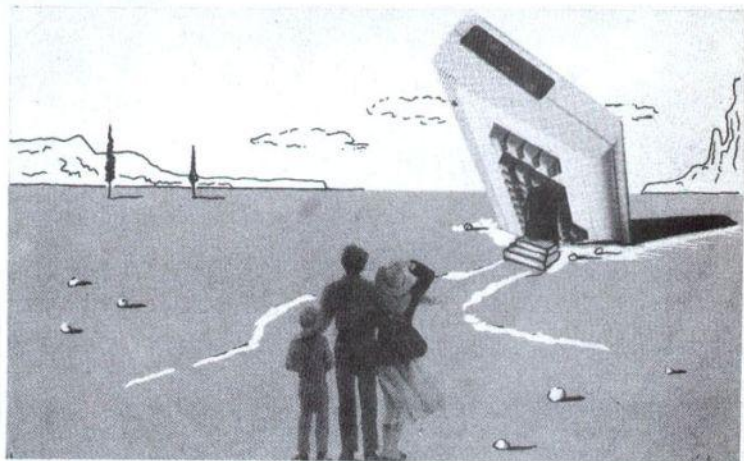
INTRODUZIONE AL KERNAL

Che cos'è il Kernal? Inizia da questo numero un viaggio nella zona di memoria più inesplorate del Computer

Una delle caratteristiche che ha decretato la rapida diffusione del computer è costituita dalla loro possibilità di conservare informazioni, immagazzinare numeri e, più in generale, di memorizzare dati. Naturalmente, affinché tutto il sistema possa gestire convenientemente queste grandi risorse è necessario che la stessa sia organizzata efficientemente in modo da renderne l'accesso il più semplice ed efficace possibile.

Zone di memoria

Per questo motivo la memoria di ogni computer è suddivisa in zone distinte, ognuna con specifiche funzioni. Con particolare riferimento al Commodore 64 (ma il discorso è valido, apportando piccole variazioni, anche per il Vic 20), queste diverse zone, nella configurazione standard, sono le seguenti (vedere al riguardo anche lo schema di figura 1):

- dalla locazione 0 alla 1023 si trova 1 Kbyte di RAM per le variabili di sistema (locazioni che il 64 usa per immagazzinare importanti valori ed indirizzi per le sue operazioni);
- da 1024 a 2047 un altro Kbyte di RAM per la grafica (e precisamente 1000 byte per la mappa video più 24 byte per gli sprite);
- da 2048 a 40959 abbiamo la RAM disponibile per i programmi Basic dell'utente;
- da 40960 a 49151 troviamo 8 Kbyte di RAM dedicati all'interprete Basic;
- da 49152 a 53247 sono situati 4 Kbyte di RAM non accessibili da Basic e perciò particolarmente idonei ad ospitare programmi in linguaggio macchina;
- da 53248 a 57343 è mappato l'Input /Output (zona usata dal 64 per comunicare con le altre periferiche come floppy disk, stampanti, ecc.);
- da 57344 a 65535 sono presenti gli 8 Kbyte di RAM del KERNAL.

A differenza delle altre zone di memoria, gli ultimi 8K del Kernal non hanno quasi mai avuto descrizioni sufficienti.

Che cos'è il Kernal?

Ebbene, il Kernal non è altro che il sistema operativo del C64, tutto l'insieme dei programmi che hanno il compito di facilitare l'uso delle risorse del computer da parte degli utenti. Non c'è bisogno di conoscere la particolare struttura hardware del calcolatore: è sufficiente ordinare al sistema operativo di occuparsi del problema.

Per fare un esempio, senza sistema operativo non ci sarebbe possibile colloquiare col computer mediante la semplice pressione di alcuni tasti a riceverne le risposte su video.

Perciò tutto l'input, l'output e la gestione della memoria è controllato dal Kernal. Inoltre, i vari programmi in linguaggio macchina che esso usa per effettuare le proprie operazioni sono accessibili anche dall'esterno (programmando in Basic o in linguaggio macchina) mediante la "tabella dei salti" situata negli ultimi 256 byte (pagina) della memoria, questa "tabella dei salti" non è altro che una zona di memoria dove sono contenuti, uno per uno, i vari indirizzi delle sottoprocedure a cui bisogna fare riferimento per ottenere l'effetto desiderato.

Per fare un esempio in Basic, se noi sappiamo che alla linea 1000 c'è il salto ad una routine che ha il compito di attendere la pressione di un tasto dalla tastiera, basterà fare riferimento con un GOTO o un GOSUB a questa linea per ottenere l'effetto descritto.

Poiché, molte volte, è necessario comunicare anche alcuni parametri (si pensi al LOAD del Basic che ha bisogno del nome del programma e del numero di periferica), prima del salto alla routine bisognerà inviare tali dati mediante l'utilizzo di appositi registri (locazioni di memoria con compiti particolari) che saranno di volta in volta specificati.
Gli stessi registri, poi, al ritorno dalla routine, potranno contenere eventuali parametri inviati dal sistema all'utente (si pensi ai vari messaggi di errore che a volte compaiono sul video).

Basterà — in caso di difficoltà — tener presente che, per usare il Kernal, si deve abbandonare la mentalità del Basic per entrare in quella del linguaggio macchina, dove qualsiasi scambio di dati è effettuato in base a valori introdotti in opportune locazioni di memoria e non a variabili o a caratteri digitati sullo schermo.

Le routine del Kernal

Bisogna premettere che, per distinguere una sottoprocedura dall'altra (in tutto ce ne sono 39), ognuna ha un proprio nome (che è quello riportato sulla Guida di Riferimento del Programmatore). Saranno trattate solamente quelle che si ritengono più importanti ai fini della programmazione.

La prima sottoprocedura che esaminiamo è quella chiamata SCREEN, a cui si accede mediante un salto alla locazione 65517 ($FFED). Scopo di questa routine è quello di ritornare al formato dello schermo (righe per colonne), rivelandosi perciò particolarmente utile per facilitare la compatibilità dei programmi scritti per i computer COMMODORE.

Il suo uso è molto semplice: lavorando in linguaggio macchina, basterà semplicemente saltare al suo indirizzo di chiamata con un JSR e, al ritorno, il registro indice X conterrà il numero delle colonne, mentre il registro Y quello delle righe; lavorando invece in Basic, bisognerà saltare con una SYS al suo indirizzo di chiamata, dopodiché la locazione 781 (che simula il registro X da Basic) conterrà il numero delle colonne e la locazione 782 (che simula il registro Y) conterrà il numero delle righe.

65535		FFFF
	KERNAL	E000
57344		E000
57343		DFFF
	INPUT-OUTPUT	
53248		D000
53247		CFFF
	RAM BUFFERS	
49152		C000
49151		BFFF
	INTERPRETE BASIC	
40960		A000
40959		9FFF
	RAM BASIC	
2048		0800
2047		07FF
	RAM GRAFICA	
1024		0400
1023	VARIAB. DI SIST.	03FF
0		0000

Figura 1: Mappa di memoria del Commodore 64

Chiariamo il tutto con un esempio. Supponiamo di voler scrivere un programma che possa girare sia sul Vic 20 che sul Commodore 64 e che, ad un certo punto, si abbia bisogno di disabilitare il tasto RUN/STOP.

Come forse già si saprà, per ottenere questo effetto basta digitare un POKE 788,52 per il C64 ed un POKE 788,194 per il VIC 20. Il problema che si pone, perciò, è quello di far eseguire o l'una o l'altra POKE a seconda della macchina su cui il programma sta girando. Con l'uso della routine SCREEN del Kernal, il problema è presto risolto. Giacché lo schermo del C64 è costituito da 40 colonne per 25 righe, mentre il Vic 20 ha uno schermo di 23 per 22, basterà saltare a questa routine e, al ritorno, controllare il numero delle colonne (o delle righe) ed agire di conseguenza.

In Basic, il programma sarà il seguente:

```
10 SYS 65517
20 IF PEEK (781) = 40 THEN POKE
788,52: GOTO 40
30 POKE 788,194
40 .....
```

Mentre in linguaggio macchina:

```
02A7 20   ED   FF  JSR  $FFED
02AA E0   28       CPX  #$28
02AC D0   05       BNE  $02B3
02AE A9   34       LDA  #$34
02B0 4C   B5   02  JMP  $02B5
02B3 A9   C2       LDA  #$C2
02B5 8D   14   03  STA  $0314
02B8 00           BRK
```

Commentiamoli brevemente. Per il programma Basic, la linea 10 esegue il salto alla routine SCREEN, dopodiché la linea 20 controlla il valore della locazione 781. Se esso è uguale a 40 (caso del C64) allora si effettua la relativa POKE e si salta alla linea 40 dove continua il programma. Nel caso in cui questo valore non fosse 40 (e allora sarebbe il 23 del Vic 20), la linea 20 non viene eseguita, si effettua la POKE della linea 30 (che è quella relativa al Vic 20) e si continua normalmente il programma.
Per il listato assembly (valido sia per il Vic 20 che per il C64), la prima istruzione esegue il salto alla routine SCREEN, dopodiché si compara il registro X col valore $28 (40 in decimale).

Se questi due sono uguali (caso del C64), allora si carica l'accumulatore col valore $34 (52 in decimale) e si salta alla locazione $0314 (788 in decimale) il valore presente in quel momento nell'accumulatore, disabilitando così il tasto STOP per il C64.

Nel caso in cui il confronto di $02AA fallisse (caso del Vic 20), allora ci sarebbe il salto a $02B3 dove si carica l'accumulatore col valore $C2 (194 in decimale) e lo si memorizza in $0314 (come già detto, 788 in decimale), disabilitando così il tasto STOP per il Vic 20. Il listato, poi, termina con l'istruzione BRK, da rimuovere nel caso in cui il programma continua.

Concludo questo mio articolo facendo notare, dall'esame della piccola routine presentata, quale possa essere la potenza del Kernal, rimandandovi nello stesso tempo ai miei prossimi scritti coi quali continueremo il nostro viaggio nel suo interno alla ricerca di routine ancor più utili e potenti.

M.C.D. & M.C.M. FRA DUE NUMERI

Calcolare il Massimo Comune Multiplo e il Minimo Comune Divisore fra due numeri è forse una delle prime cose che abbiamo imparato alle scuole medie. Il metodo insegnato è quello di scomporre i due numeri in fattori primi e poi, seguendo un certo criterio, valutare l'MCM e l'MCD. Questo, però, è un metodo alquanto scomodo, soprattutto quando lo si vuole "insegnare" ad un computer.

Come si fa infatti a trovare l'MCM o l'MCD fra due numeri di sei o più cifre, usando il metodo della scomposizione in fattori primi? È una cosa molto lunga, esiste però, per fortuna, un altro metodo che, guarda caso, è fatto apposta per il computer, e nello stesso momento risulta essere velocissimo (provare il programma per credere).

Questo metodo è l'"algoritmo delle divisioni successive". Si tratta insomma di compiere solamente alcune divisioni. Chiamiamo i due numeri dati AX e BX. Troviamo allora un QX che è il quoziente di AX e BX, cioè QX = AX/BX. Chiamiamo RX il resto di questa divisione.

Se il resto è diverso da zero, si prosegue con le divisioni, e bisogna dividere BX per QX, ottenendo un altro QX e un altro RX e poi ancora dividere il primo QX per il secondo QX, e così via, fino a quando non si ottiene resto uguale a zero.

In questo caso il penultimo divisore, cioè il QX che abbiamo diviso per il secondo QX, rappresenta il Massimo Comune Divisore. Non è il caso questo di dimostrare quanto detto: provate a far girare il programma e verificate che le soluzioni siano esatte. Se poi vogliamo trovare anche il Minimo Comune Multiplo, non dobbiamo fare altro che moltiplicare i nostri due numeri di partenza fra loro e dividere il risultato per il Massimo Comune Divisore.

Tutto questo avviene in un tempo brevissimo, anche per numeri di molte cifre.

Giovanni Bellù

```
110 REM    M.C.D   &  M.C.M. FRA DUE NUMERI
120 REM
130 REM    GIOVANNI BELLU' SOFTWARE 1984
190 :
200 REM SFONDO BLU PER C= 64
210 POKE53280,6
220 POKE53281,6
230 PRINT"█":REM CLR HOME
240 PRINT"◥":REM CYN.
250 PRINT"█ C A L C O L O - D E L "
260 PRINT"██MASSIMO COMUNE DIVISORE"
270 PRINT"██MINIMO  COMUNE MULTIPLO"
280 PRINT"█FRA DUE NUMERI DATI A,B"
290 INPUT"███INTRODUCI A,B";A,B
300 IFB=0ORA=0THENRUN
310 AX=A:BX=B
320 REM
330 REM
340 REM ALGORITMO DELLE DIVISIONI SUCCESSIVE
350 REM
360 QX=INT(AX/BX)
370 RX=AX-BX*QX:IFRX=0THEN460
380 IFBX/RX=INT(RX/BX)THEN400
390 AX=BX:BX=RX:GOTO360
400 PRINT"█M.C.D.█=";RX
410 REM CALCOLO MCM
420 MM=A*B/RX
430 PRINT"█M.C.M.█=";MM
440 PRINT
450 END
460 PRINT"█M.C.D.█=";BX
470 MM=A*B/BX
480 PRINT"█M.C.M.█=";MM
490 PRINT
READY.
```

TUNNEL

Scopo del gioco: evitare mine e astronavi che vi piombano addosso sparando. Potete sparare sia alle astronavi che alle mine e non potete uscire fuori dal tunnel.

Ostacoli: mine, non si muovono ma sono micidiali se entrate in collisione con esse; astronavi, piombano volando verso di voi e lanciano raffiche di missili.

Se un missile o un'astronave entra in collisione vi fa perdere una delle 3 vite. Si guadagnano mille punti per ogni astronave e 100 per ogni mina. Non potete bloccare i missili delle astronavi.

Il gioco prosegue dopo aver eliminato il primo squadrone di astronavi con altri squadroni. Il gioco termina quando perdete le tre vite.

Andrea Campione

```
100 REM*        TUNNEL
110 REM* PER VIC 20 INESPANSO
120 REM*
130 REM*  DI ANDREA CAMPIONE
170 :
171 PRINT"USA I TASTI"
172 PRINT"N (=SINISTRA)"
173 PRINT"M (=DESTRA)"
174 PRINT"Z (=SPARO)"
175 PRINT"PREMI UN TASTO"
176 GETA$:IFA$=""THEN176
180 POKE36879,110:POKE36878,15
190 POKE36869,255
200 FORI=7168TO8185:POKEI,0:NEXT
210 FORI=7176TO7183:READA:POKEI,A:NEXT
220 FORI=7184TO7191:READB:POKEI,B:NEXT
230 FORI=7192TO7199:READC:POKEI,C:NEXT
240 FORI=7200TO7207:READD:POKEI,D:NEXT
250 FORI=7208TO7215:READE:POKEI,E:NEXT
260 V=3:P=0
270 PRINT"":S1=2:S2=2:S3=2:S4=2
280 FORX=1TO21
290 PRINT"      D          D"
300 NEXTX
310 AS=8174:NE=7688:SP=NE+22
320 FORX=1TO50
330 H=INT(506*RND(1))
340 POKE7680+H,4
350 NEXTX
360 PRINT"SCORE="P" LIVES="V:POKE36876,0:GOSUB650
370 IFPEEK(AS-22)=2ORPEEK(AS-22)=4THENGOSUB980:GOTO270
380 IFAS<7680THENAS=8174
390 IFV=0THEN890
400 GOSUB650
410 IFNE>8185THENNE=7688
420 POKENE,0:POKENE+23,0:POKENE+48,0:POKENE+72,0:GOSUB650
430 NE=NE+22
440 POKENE,S1:POKENE+23,S2:POKENE+48,S3:POKENE+72,S4:GOSUB650
450 POKEAS,0:POKEAS-22,1:AS=AS-22
460 FORNU=1TO5
470 IFSP>8185THENGOSUB740
480 IFPEEK(SP+22)=1THENV=V-1:POKESP,0:POKESP+22,3:GOSUB820
```

```
490 POKESP,0:POKESP+22,5:SP=SP+22:GOSUB650
500 GOSUB650
510 POKE36877,200
520 IFS1=0ANDS2=0ANDS3=0ANDS4=0THEN270
530 NEXTNU
540 FORZX=1TO5
550 IFW<7680THENW=7679
560 IFPEEK(W-22)=4THENP=P+100:POKEW,0:POKEW-22,3:W=7679
570 IFW-22=NEANDS1<>0THENS1=0:P=P+1000:POKEW,0:POKEW-22,3:W=7679
580 IFW-22=NE+23ANDS2<>0THENS2=0:P=P+1000:POKEW,0:POKEW-22,3:W=7679
590 IFW-22=NE+48ANDS3<>0THENS3=0:P=P+1000:POKEW,0:POKEW-22,3:W=7679
600 IFW-22=NE+72ANDS4<>0THENS4=0:P=P+1000:POKEW,0:POKEW-22,3:W=7679
610 POKE36876,253:GOSUB650
620 POKEW-22,5:POKEW,0:W=W-22:GOSUB650
630 NEXTZX
640 GOTO360
650 IFPEEK(197)=36THEN690
660 IFPEEK(197)=28THEN710
670 IFPEEK(197)=33THEN730
680 RETURN
690 IFPEEK(AS+1)=4THENRETURN
700 POKEAS,0:POKEAS+1,1:AS=AS+1:RETURN
710 IFPEEK(AS-1)=4THENRETURN
720 POKEAS,0:POKEAS-1,1:AS=AS-1:RETURN
730 POKEW,0:W=AS-22:RETURN
740 K=INT(100*RND(1))
750 IFK<50THENRETURN
760 J=INT(4*RND(1))
770 IFJ=0ANDS1<>0THENSP=NE+22
780 IFJ=1ANDS2<>0THENSP=NE+45
790 IFJ=2ANDS3<>0THENSP=NE+70
800 IFJ=3ANDS4<>0THENSP=NE+94
810 RETURN
820 POKE36877,128:POKE36879,45
830 FORM=15TO0STEP-1
840 POKE36878,M
850 FORX=1TO100:NEXTX
860 NEXTM
870 POKE36878,15:POKE36879,110
880 RETURN
890 POKE36869,240
900 POKE198,0:PRINT"[illegible]FINE GIOCO"
910 POKE36878,0:PRINT"PUNTI="P
920 END
930 DATA24,60,126,231,231,255,129,189
940 DATA31,32,64,159,249,2,4,248
950 DATA128,48,4,8,18,65,0,129
960 DATA126,189,219,231,231,219,189,126
970 DATA129,129,129,129,129,129,129,129
980 V=V-1:POKEAS,0:POKEAS-22,3:GOSUB820:RETURN

READY.
```

MUSIC EDITOR

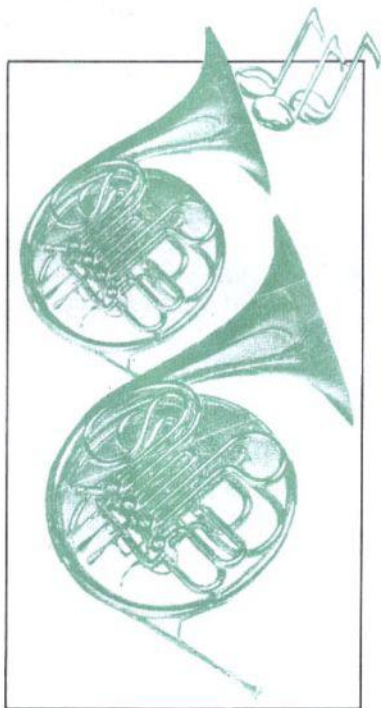
Fare musica con il Commodore 64 è quasi un obbligo per chi possiede tale computer. Non ci si deve accontentare dei soliti rumori, ma bisogna realizzare dei "suoni" un po' più... complessi: della musica. Per fare questo il Commodore 64 mette a disposizione ben tre voci, e quindi, in pratica, tre note che possono suonare nello stesso momento.

Chi ha provato a creare della musica senza l'aiuto di alcun programma, saprà certamente quale duro lavoro bisogna fare nel cercare i valori delle note nella apposita tabella che si trova nelle appendici del libretto di istruzioni e nell'introdurli in un programma che li legge e, successivamente, li "suona".

Cominciamo, però, dall'inizio: ci troviamo davanti ad uno spartito di musica e dobbiamo introdurlo in qualche modo nel computer. Per fare questo si incontrano non poche difficoltà, prima fra tutte è quella che le note non sono della stessa durata. Come fare allora per suonare insieme delle note la cui durata non è la stessa?

Vi sono diversi metodi, tra cui quello di considerare per ogni nota la sua durata e poi affidare tutto il resto al computer. Un altro metodo, quello adottato dal mio programma, è di suddividere le note in base alla nota di minor durata.

Per essere più chiari, se la nota più breve è di un sedicesimo, bisognerà suddividere l'intero spartito in sedicesimi, così che la nota di durata, per esempio, di un ottavo andrà ripetuta due volte, quella di un quarto quattro volte e così via. In questo modo si "consuma" più memoria, ma le correzioni, (o le varianti!), risultano più facili.

Lo stesso procedimento di lettura e di conversione in musica dei dati viene in tal modo semplificato, come si può notare dal secondo listato, di cui parleremo dopo.

Il programma

Il music editor proposto ha la funzione di "tradurre" le note in valori numerici, archiviarle, stamparle e suonarle. All'inizio bisogna attendere qualche secondo perché il programma calcoli i valori di tutte le note delle otto ottave, avendo a disposizione solamente le frequenze di ogni nota dell'ultima ottava (linee di data da 390 a 500). Il metodo usato è molto semplice ed è quello indicato nella "Guida di riferimento per il programmatore" (appendice "O" a pag. 75). Esso consiste nel fatto che ogni nota di una ottava ha come frequenza quella della stessa nota della ottava successiva, divisa però per due.

Basta quindi avere i valori delle frequenze dell'ultima ottava per poter ottenere mediante la semplice divisione per due, tutte le frequenze delle otto ottave. Si può vedere questo procedimento dalla linea 560 alla linea 620.

Per calcolare i valori da "Pokare" per avere musica, basta sapere che il primo (valore alto) è dato dall'intero della divisione della frequenza per 256 e che il secondo (valore basso) è semplicemente il resto.

Questi valori vengono calcolati dalla linea 640 alla linea 740.

Quando il computer avrà finito di calcolare tali valori, farà apparire sullo schermo un menú con sei opzioni, più una per uscire dal programma. Analizziamo ora una per una le sei opzioni.

Opzione 1: INPUT nuovi dati

Una volta premuto il tasto "1" per la scelta della prima opzione verrà cancellato il video e sullo schermo apparirà la scritta "introduci le note (nota-ottava)" e sotto "Tempo 1" ed il cursore su una linea.

Bisogna scrivere la nota: do, re, mi, fa, sol, la, si, premere Return e poi premere un tasto compreso fra uno e otto per indicare l'ottava. Il cursore si sposterà su una linea più a destra.

La prima linea indicava la prima voce, la seconda indica la seconda e l'ultima indicherà la terza voce. Se nella musica introdotta non c'è bisogno di tre voci, basta premere Return e verrà visualizzata una freccetta verso l'alto (questo può anche servire se all'interno della musica c'è una pausa).

Ovviamente, la possibilità di introdur-

re i DIESIS (i tasti neri, per chi non se ne intende!) non manca, e basta mettere "#" dopo la nota, cioè: do#, re#, fa#, sol#, la#.

Non è possibile introdurre i Bemolle con un simbolo particolare: bisogna farli diventare DIESIS della nota precedente. Per esempio, il Re Bemolle diventa DO*, cioè DO DIESIS.

Per gli esperti di musica questo è un grave errore teorico, anche se "praticamente", cioè agli effetti della frequenza della nota e dell'ascolto, non cambia assolutamente niente. Noi, strimpellatori con il computer ci accontentiamo di questo! I teorici si accontentino con noi...

Una volta introdotte le note e le ottave per le tre voci, la scritta "Tempo 1" cambia in "Tempo 2" e così di seguito fino al massimo (modificabile), indicato nelle istruzioni DIM di linea 270 e 280, di cento tempi, cioè di cento sedicesimi, ottavi, quarti ecc. a seconda di come avete suddiviso lo spartito in base alla nota di minor durata. Nell'introduzione delle note, sono disponibili alcune comodità, che per altro sono essenziali per chi vuole introdurre spartiti complicati o per chi teme l'errore.

Primo: premendo F1 vengono visualizzate al di sotto delle linee solite, le note del tempo precedente, e si può cambiarle come si vuole, come se si dovessero introdurle di nuovo. Se per esempio siamo nel tempo 9 e vogliamo tornare... indietro nel tempo, premendo F1 andremo nel tempo 8, premendo ancora F1 nel tempo 7 e così via fino al tempo 1 o fino a quando troviamo un tempo da correggere, per cui dobbiamo riscrivere le note e le ottave per tutte e tre le voci di quel tempo.

Secondo: premendo F3 si ottiene l'effetto opposto a quello che si ottiene premendo F1: cioè si va avanti, fino all'ultimo tempo introdotto. Se l'ultimo tempo introdotto è l'8, per esempio, e siamo andati nel tempo 3 con F1, se vogliamo continuare l'introduzione delle note, dobbiamo quindi premere F3 fino a quando si arriva nel tempo 9, il primo libero. Più in là non si può andare.

Terzo: premendo F5 si può inserire una linea. Se per esempio ci siamo dimenticati di introdurre un tempo (il quarto), dobbiamo usare F1 o F3 fino a quando siamo al quarto tempo. Premendo allora F5, i tempi dal quarto in poi verranno "shiftati" in su (cioè il quarto diventa quinto, il quinto sesto e così via) e nel quarto verrà introdotto in ogni voce il RETURN, cioè la freccetta verso l'alto, che indica, come detto prima, una nota nulla.

Possiamo a questo punto o lasciare le note nulle o introdurre come al solito le note del quarto tempo, e poi proseguire con l'introduzione delle altre note.

Quarto: premendo F7 si ottiene l'effetto opposto all'opzione data da F5, cioè si cancella un tempo. Se per esempio vogliamo cancellare il quinto tempo perché è di troppo, andiamo con F1 o F3 al tempo 5 e premiamo F7: al posto del quinto tempo verrà messo il sesto, al posto del sesto il settimo e così via fino all'ultimo tempo introdotto.

Quinto: quando si è finito di introdurre lo spartito, (o si è stanchi di introdurlo!), premendo F8, cioè SHIFT + F7, verrà visualizzata una domanda che chiede se si vuole memorizzare le note introdotte fino a quel momento. Se viene risposto di no, cioè si preme "N", si torna alle opzioni principali. Per il caso affermativo vedere l'opzione 6, in cui viene spiegata la tecnica di memorizzazione e la successiva rilettura dei dati mediante un qualsiasi programma che debba usarli.

Opzione 2: Modifiche Dati

Per usare questa opzione bisogna aver già introdotto dei dati oppure averli letti da disco mediante l'opzione 3.

Una volta scelta questa opzione, bisogna agire in modo analogo alla opzione 1 e si può quindi correggere, cancellare, inserire o introdurre nuove note. Per uscire premere, come per l'opzione 1 il tasto funzione numero 8.

Opzione 3: Lettura Dati

Bisogna introdurre il nome di un file precedentemente registrato mediante cui si può modificare, memorizzare o ascoltare i dati letti.

Opzione 4: Output su stampante

Vi sono due opzioni più una terza per tornare alle opzioni principali. La prima: "Sono note" stampa i simboli delle note (do, re, sol#, si, ecc.) e l'ottava. Su ogni linea vi sono le note delle tre voci del tempo indicato dalla linea. La seconda: "Solo dati per Poke note" stampa i valori numerici, quelli da "Pokare" per ottenere musica.

Vengono stampati, per ogni voce, il valore alto ed il valore basso della frequenza della nota in questione. Ogni linea stampata è numerata: il numero indica il tempo di quella linea.

Opzione 5: Ascolto Musica

Questa opzione inizia alla linea 2980. Bisogna attendere un po' di tempo all'inizio perché il computer traduce i simboli delle note introdotte, nei numeri corrispondenti alla frequenza di ciascuna nota.

Verrà quindi suonata la "nostra" musica usando, ovviamente, le tre voci. Quella che sentiremo è solamente una musica indicativa: infatti si può modificare il tipo di suono e, ovviamente la durata. Per quest'ultima si può modificare il loop di ritardo di linea 3370.

Alla fine si torna automaticamente alle opzioni principali.

Opzione 6: Memorizzazione

Usando questa opzione si possono memorizzare su disco i dati introdotti con l'opzione 1 o 2.

Bisogna dare il nome del file nel quale verranno memorizzati i dati. Attenzione assegnando il nome di un file precedentemente memorizzato, questo verrà cancellato!! Vengono registrati due file. Il secondo avrà lo stesso nome del primo, con l'aggiunta all'inizio di una freccetta

verso l'alto. Vediamo a che servono e come si leggono questi due file.

Il primo, che avrà esattamente il nome da noi introdotto, contiene i dati delle note così come li introduciamo, insieme all'ottava, così come compare sulla stampante scegliendo la prima opzione. (DO2, MI4, SOL#6 ecc.). Sono registrati sequenzialmente, cioè, prima la prima voce del tempo uno, poi la seconda, quindi la terza, e poi ancora la prima voce del tempo 2 e così via. Alla fine per indicare che i dati sono finiti viene memorizzato "Fine".

Tale file serve soprattutto per il programma pubblicato perché sono questi i dati che il computer va a rileggere con l'opzione 3.

Il secondo file è invece quello utile per altri scopi perché contiene tutti i valori per ogni nota da "pokare", i dati cioè della tabella di cui si diceva all'inizio. Questi dati sono memorizzati come segue:

valore alto, valore basso voce 1 tempo 1
valore alto, valore basso voce 2 tempo 1
valore alto, valore basso voce 3 tempo 1
valore alto, valore basso voce 1 tempo 2
e così via per tutti i tempi della musica memorizzata.

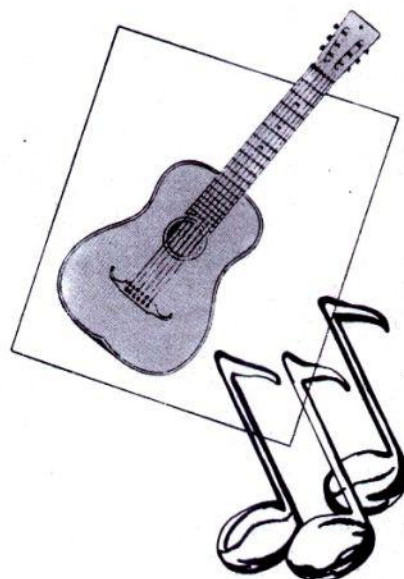
Il programma numero 2 è un esempio di come si deve fare per leggere i dati generati in tal modo e quindi "suonarli". Ovviamente è necessario aver già memorizzato almeno un file con delle note su dischetto.

Nota bene **(Per chi non possiede floppy o per chi vuole utilizzare la musica così composta in un proprio programma).**

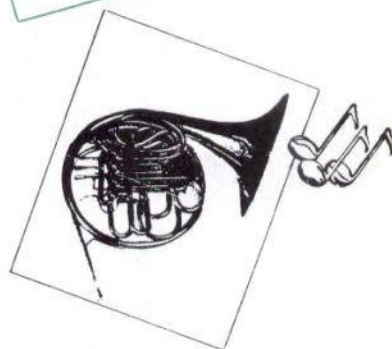
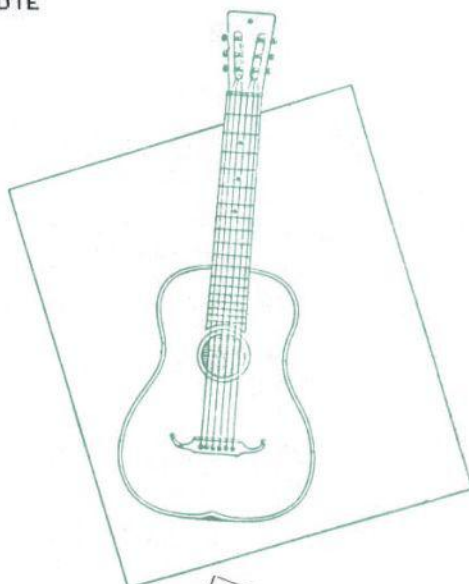
I valori memorizzati in questo secondo file potrebbero venir messi nelle istruzioni DATA, così che il Programma che li utilizza non è costretto ogni volta a leggerseli da disco. Per fare questo è utile stampare tali valori sulla stampante usando la seconda opzione dell'opzione 4. Nella figura viene mostrato un esempio di output su stampante Poke autom. Provate ad inserire quei dati nel programma e... buon ascolto.

	VOCE 1	VOCE 2	VOCE 3
1 ->	FA3	FA4	↑
2 ->	↑	↑	↑.
3 ->	FA3	FA4	↑
4 ->	FA3	FA4	↑
5 ->	FA3	FA4	↑
6 ->	↑	↑	↑
7 ->	FA3	FA4	↑
8 ->	LA#3	RE4	LA#4
9 ->	LA#3	RE4	LA#4
10->	LA#3	RE4	LA#4
11->	LA#3	RE4	LA#4
12->	↑	↑	↑
13->	LA#3	RE4	LA#4
14->	LA#3	RE4	LA#4
15->	LA#3	RE4	LA#4
16->	LA#3	RE4	LA#4
17->	FA3	FA4	DO4
18->	FA3	FA4	DO4
19->	FA3	FA4	DO4
20->	FA3	FA4	DO4
21->	↑	↑	↑
22->	FA3	FA4	DO4
23->	FA3	FA4	DO4
24->	FA3	FA4	DO4
25->	FA3	FA4	DO4
26->	RE3	FA4	FA5
27->	RE3	FA4	FA5
28->	RE3	FA4	FA5
29->	RE3	FA4	FA5
30->	FA3	FA4	FA5
31->	FA3	FA4	FA5
32->	FA3	RE5	↑
33->	FA3	RE5	↑
34->	RE3	LA#4	↑
35->	RE3	LA#4	↑
36->	RE3	↑	↑
37->	RE3	LA#4	↑
38->	LA#2	RE5	↑
39->	LA#2	RE5	↑
40->	LA#2	RE5	↑
41->	RE3	LA#4	RE2
42->	RE#2	RE#3	SOL4
43->	RE#2	RE#3	SOL4
44->	RE#2	RE#3	SOL4
45->	RE#2	RE#3	SOL4
46->	DO3	SOL4	RE#5
47->	DO3	SOL4	RE#5
48->	DO3	SOL4	RE#5
49->	DO3	SOL4	RE#5
50->	DO3	SOL4	RE#5
51->	DO3	SOL4	RE#5
52->	DO3	SOL4	RE#5
53->	DO3	SOL4	RE#5
54->	FA3	FA4	DO5
55->	FA3	FA4	DO5
56->	FA3	FA4	DO5
57->	FA2	FA3	LA4
58->	LA#2	RE4	LA#4
59->	LA#2	RE4	LA#4
60->	LA#2	RE4	LA#4
61->	LA#2	RE4	LA#4
62->	↑	↑	LA#4
63->	↑	↑	LA#4
64->	↑	↑	LA#4
65->	↑	↑	LA#4
66->	↑	↑	LA#4
67->	↑	↑	LA#4
68->	↑	↑	LA#4
69->	↑	↑	LA#4

TOTALE TEMPI 69

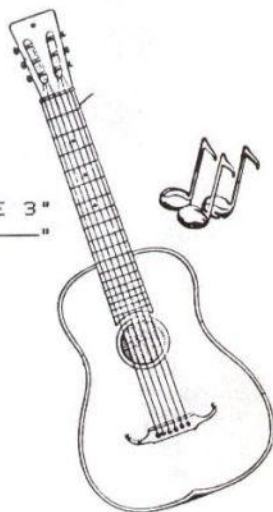
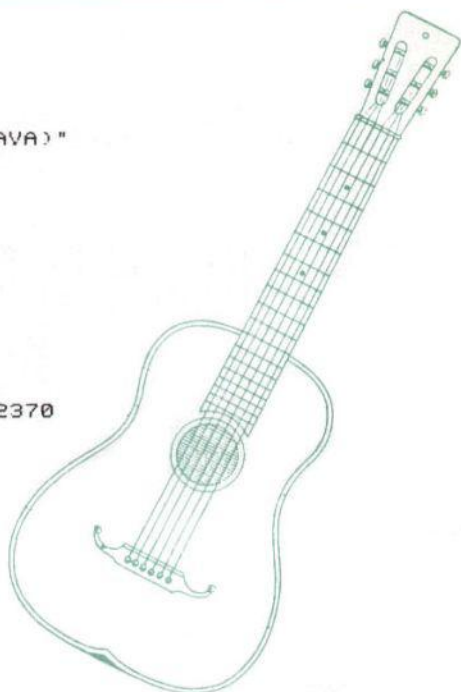
```
100 REM     LETTURA  DATI MEMORIZZATI
110 REM
120 REM              DAL
130 REM
140 REM     M U S I C - E D I T O R
150 REM
160 REM
170 REM  GIOVANNI BELLU' SOFTWARE 1984
180 REM
220 REM
230 REM
240 PRINT"■"
250 POKE53280,6:POKE53281,6
260 PRINT"◣"
270 DIMNT(100,3,2)
280 INPUT"NOME FILE";NF$
290 OPEN15,8,15,"I0":GOSUB570
300 OPEN2,8,2,"0:↑"+NF$+",S,R":GOSUB570
310 N=N+1
320 FORJ=1TO3
330 FORI=1TO2
340 INPUT#2,NT(N,J,I)
350 IFNT(N,J,I)=-1THEN390
360 NEXT
370 NEXT
380 GOTO310
390 J=4:I=3
400 NEXT
410 NEXT
420 N=N-1:CLOSE2:CLOSE15
430 PRINT"■■ OK LETTI"N" TEMPI"
440 REM
450 REM****************************
460 REM*METTERE LE LINEE 510 E 520*
```

```
470 REM* SE SI VOGLIONO VEDERE I  *
480 REM* DATI LETTI DAL PROGRAMMA *
490 REM****************************
500 REM
510 REM FORK=1TON:FORJ=1TO3:FORI=1TO2
520 REM PRINTNT(K,J,I);:NEXT:PRINT"   ";:NEXT:PRINT:NEXT
530 GOTO610
540 REM
550 REM LETTURA MESSAGGI DI LISTE
560 REM
570 INPUT#15,A$,B$,C$,D$
580 IFA$="00"THENRETURN
590 PRINT"  ERRORE SU DISCO "
600 PRINTA$" "B$,C$D$
610 REM
620 REM RESET SUONO
630 REM
640 FORK=54296TO54360:POKEK,0:NEXT
650 POKE54296,15:REM VOLUME
660 POKE54277,255:POKE54278,255:POKE54276,33
670 POKE54284,255:POKE54285,255:POKE54283,33
680 POKE54291,255:POKE54292,255:POKE54290,33
690 H1=54273:H2=54280:H3=54287
700 L1=54272:L2=54279:L3=54286
710 REM
720 REM SUONO MUSICA
730 REM
740 FORK=1TON-1
750 POKEH1,NT(K,1,1):POKEL1,NT(K,1,2)
760 POKEH2,NT(K,2,1):POKEL2,NT(K,2,2)
770 POKEH3,NT(K,3,1):POKEL3,NT(K,3,2)
780 FORJ=1TO49:NEXT:REM RITARDO
790 NEXTK
800 FORK=54296TO54360:POKEK,0:NEXT
810 PRINT" ANCORA"
820 GETA$:IFA$="S"THEN640
830 IFA$<>"N"THEN820
840 PRINT" ":N=N-1
850 INPUT"DA QUALE LOCAZIONE MEMORIZZO";LC
860 O=0:IFLC<10000ORLC>53248-N*6THEN850
870 FORK=LCTOLC+N*6STEP6:O=O+1
880 FORI=0TO2:FORX=0TO1
890 POKEK+X+2*I,NT(O,I+1,X+1);
900 NEXTX,I:PRINT:NEXT

READY.
```

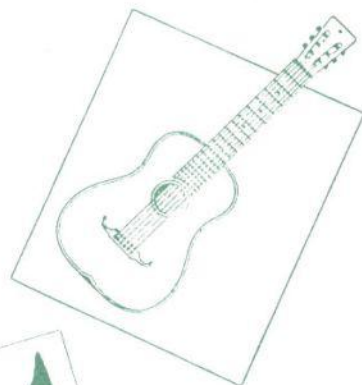
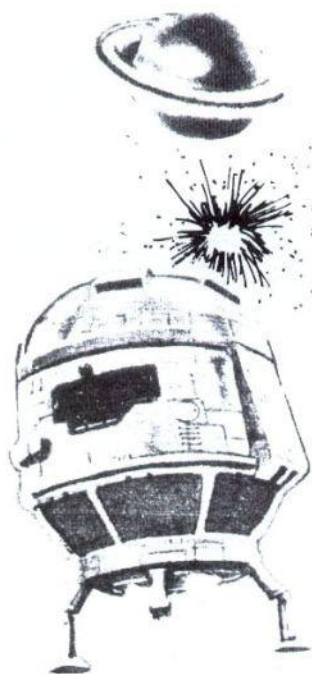
```
100 REM
110 REM      M U S I C - E D I T O R
120 REM
130 REM
```

```
140 REM  GIOVANNI BELLU' SOFTWARE 1984
150 REM
190 REM
200 PRINT"█ATTENDI UN MOMENTO"
210 REM
220 REM DIMENSIONAMENTO MATRICI
230 REM
240 DIM FR(8,12) :REM FREQUENZA
250 DIM NT(8,12,2) :REM VALORI POKE NOTE
260 DIM S$(12) :REM SIMBOLI NOTE
270 DIM N$(100,3) :REM SEQUENZA NOTE
280 DIM VL(100,3,2) :REM VALORI NOTE
290 DIM E$(3,2) :REM PER STAMPA VALORI NOTE
300 REM
310 REM
320 REM
330 REM CAMBIA COLORE DI SFONDO
340 REM
350 POKE53280,0:POKE53281,0
360 REM
370 REM FREQUENZA NOTE ULTIMA OTTAVA
320 REM
390 DATA34334:REM C -8 = DO
400 DATA36376:REM C#-8 = DO#
410 DATA38539:REM D -8 = RE
420 DATA40830:REM D#-8 = RE#
430 DATA43258:REM E -8 = MI
440 DATA45830:REM F -8 = FA
450 DATA48556:REM F#-8 = FA#
460 DATA51443:REM G -8 = SOL
470 DATA54502:REM G#-8 = SOL#
480 DATA57743:REM A -8 = LA
490 DATA61176:REM A#-8 = LA#
500 DATA64814:REM B -8 = SI
510 REM
520 REM LETTURA FREQUENZE
530 REM
540 FORK=1TO12:READFR(8,K):NEXT
550 REM
560 REM CALCOLO FREQUENZE ALTRE OTTAVE
570 REM
580 FORK=7TO1STEP-1
590 FORJ=1TO12
600 FR(K,J)=INT(FR(K+1,J)/2)
610 NEXT
620 NEXT
630 REM
640 REM CALCOLO VALORI NOTE PER POKE
650 REM
660 FORK=1TO8
670 FORJ=1TO12
680 A=INT(FR(K,J)/256)
```

```
690 B=FR(K,J)-256*A
700 NT(K,J,1)=A
710 NT(K,J,2)=B
720 NEXT
730 NEXT
740 REM
750 REM RESET SUONO
760 REM
770 FORK=54296TO54360:POKEK,0:NEXT
780 REM
790 REM INPUT NOTE
800 REM
810 DATADO,DO#,RE,RE#,MI,FA,FA#,SOL,SOL#,LA,LA#,SI
820 FORK=1TO12:READS$(K):NEXT
830 FORK=1TO39:C$=C$+CHR$(32):NEXT
840 GOTO1750
850 PRINT"■"
860 PRINT"■■ INTRODUCI LE NOTE ■(NOTA-OTTAVA)"
870 N=1:N1=N
880 REM
890 REM F8 = FINE INTRODUZIONE NOTE
900 REM
910 PRINT"■■■■TEMPO■    ■■■■"N:FORJ=1TO3
920 W$="■■■■■":TB=(J-1)*10:TC=TB:Q$="■—■":Q1$="■—":B$=""
930 PRINTW$TAB(TC)"———— —"
940 PRINTW$TAB(TB)Q$
950 FORK=1TO50:GETA$:IFA$=""THENNEXT:GOTO1080
960 IFASC(A$)=20THENTB=TB-1:GOSUB1230:IFTB<0THENTB=0:NEXT:GOTO1080
970 IFA$="■"THEN1910:REM F1
980 IFA$="■"THEN2060:REM F3
990 IFA$="■"THEN3500:REM F5
1000 IFA$="■"THEN3650:REM F7
1010 IFASC(A$)=13THENK=60:NEXT:GOTO1270
1020 IFA$="#"THEN1050
1030 IFA$="■"THENK=60:J=4:NEXTK,J:GOTO1440:REM F8
1040 IFA$<"A"ORA$>"S"THENNEXT:GOTO1080
1050 IFLEN(B$)=4THEN1070
1060 B$=B$+A$:PRINTW$TAB(TC)B$:TB=TB+1
1070 NEXT
1080 PRINTW$TAB(TB)Q1$
1090 FORK=1TO50:GETA$:IFA$=""THENNEXT:GOTO940
1100 IFASC(A$)=20THENTB=TB-1:GOSUB1230:IFTB<0THENTB=0:NEXT:GOTO940
1110 IFA$="■"THEN1910:REM F1
1120 IFA$="■"THEN2060:REM F3
1130 IFA$="■"THEN3500:REM F5
1140 IFA$="■"THEN3650:REM F7
1150 IFASC(A$)=20THENTB=TB-1:GOSUB1230:IFTB<0THENTB=0:NEXT:GOTO940
1160 IFASC(A$)=13THENK=60:NEXT:GOTO1270
1170 IFA$="#"THEN1200
1180 IFA$="■"THENK=60:J=4:NEXTK,J:GOTO1440:REM F8
1190 IFA$<"A"ORA$>"S"THENNEXT:GOTO940
1200 IFLEN(B$)=4THEN1220
```

```
1210 B$=B$+A$:PRINTW$TAB(TC)B$:TB=TB+1
1220 NEXT:GOTO940
1230 PRINTW$TAB(TC)"——— —"
1240 IFLEN(B$)<2THENB$="":RETURN
1250 B$=LEFT$(B$,LEN(B$)-1):PRINTW$TAB(TC)B$
1260 RETURN
1270 REM
1280 REM CONTROLLO INPUT NOTA GIUSTA
1290 REM
1300 IFB$=""THENA$="":B$="↑":PRINTW$TAB(TC)"——— —";W$TAB(TC)B$:GOTO1410
1310 FORK=1TO12
1320 IFB$=S$(K)THENK=13:NEXT:GOTO1340
1330 NEXT:B$="":GOTO920
1340 PRINTW$TAB(TC)"——— —":PRINTW$TAB(TC)B$:PRINTW$TAB(TC+5)"■—"
1350 GETA$:IFA$<"1"ORA$>"8"THEN1350
1360 PRINTW$TAB(TC+5)"■"A$
1370 GETK$:IFK$=""THEN1370
1380 IFASC(K$)=20THEN1340
1390 IFASC(K$)<>13THEN1370
1400 PRINTW$TAB(TC+5)A$
1410 N$(N,J)=B$+A$
1420 NEXTJ:IFN<N1THENN=N1:PRINTW$"■■"C$:GOTO910
1430 N=N+1:N1=N:GOTO910
1440 REM
1450 REM FINE INTRODUZIONE NOTE
1460 PRINT"■VUOI MEMORIZZARE LE NOTE ?"
1470 GETA$:IFA$="N"THEN1750
1480 IFA$="S"THEN1500
1490 GOTO1470
1500 REM MEMORIZZAZIONE NOTE
1510 OPEN15,8,15,"I0":GOSUB3470:INPUT"■■NOME DEL FILE";NF$
1520 OPEN2,8,2,"@0:"+NF$+",S,W"
1530 FORK=1TON:FORJ=1TO3
1540 PRINT#2,N$(K,J);CHR$(13);:GOSUB3470
1550 NEXT:NEXT:PRINT#2,"FINE";CHR$(13):GOSUB3470
1560 CLOSE2
1570 PRINT"■OK- ATTENDI: ALTRO FILE"
1580 FORI=1TON1-1
1590 FORJ=1TO3:IFN$(I,J)="↑"THENO=0:X1=0:GOTO1660
1600 O$=RIGHT$(N$(I,J),1)
1610 O=VAL(O$)
1620 N$=LEFT$(N$(I,J),LEN(N$(I,J))-1)
1630 FORX=1TO12
1640 IFS$(X)=N$THENX1=X:X=13:NEXT:GOTO1660
1650 NEXT:PRINT"■■ ERRORE ":END
1660 VL(I,J,1)=NT(O,X1,1)
1670 VL(I,J,2)=NT(O,X1,2)
1680 NEXT:NEXT
1690 PRINT"■ MEMORIZZO ":OPEN2,8,2,"@:↑"+NF$+",S,W":GOSUB3470
1700 FORK=1TON1-1:FORJ=1TO3:FORI=1TO2
1710 PRINT#2,VL(K,J,I);CHR$(13);:GOSUB3470
1720 NEXT:NEXT:NEXT
```

```
1730 EE=-1:FORK=1TO2:PRINT#2,EE;CHR$(13);:GOSUB3470:NEXT:
     PRINT#2,EE;CHR$(13)
1740 GOSUB3470:CLOSE2:CLOSE15
1750 PRINT"■"
1760 PRINT"1 - INPUT NUOVI DATI"
1770 PRINT"2 - MODIFICHE DATI"
1780 PRINT"3 - LETTURA DATI"
1790 PRINT"4 - OUTPUT SU STAMPANTE"
1800 PRINT"5 - ASCOLTO MUSICA"
1810 PRINT"6 - MEMORIZZAZIONE"
1820 PRINT"7 - FINE PROGRAMMA"
1830 GETA$
1840 IFA$<"1"ORA$>"7"THEN1830
1850 Q=VAL(A$)
1860 ONQGOTO850,2210,2260,2420,2980,1440,1880
1870 GOTO1830
1880 PRINT"■■":END
1890 END
1900 REM
1910 REM SE VIENE PREMUTO F1 = INDIETRO
1920 REM
1930 REM TEMPO PRECEDENTE
1940 IFN=1THEN910
1950 N=N-1:PRINT"■■■■TEMPO■    ■■■■"N:PRINTW$C$
1960 FORK=0TO2
1970 PRINTW$"■■"TAB(K*10)"—————  —"
1980 NEXT
1990 FORK=1TO3:TB=(K-1)*10
2000 B$=LEFT$(N$(N,K),LEN(N$(N,K))-1)
2010 A$=RIGHT$(N$(N,K),1)
2020 PRINTW$"■■"TAB(TB)B$W$"■■"TAB(TB+5)A$
2030 NEXT
2040 GOTO910
2050 REM
2060 REM SE VIENE PREMUTO F3 = AVANTI
2070 REM
2080 REM TEMPO SUCCESSIVO
2090 IFN=N1THEN910
2100 IFN+1=N1THENN=N1:PRINTW$"■■"C$:GOTO910
2110 N=N+1:PRINT"■■■■TEMPO"N:PRINTW$C$
2120 FORK=0TO2:PRINTW$"■■"TAB(K*10)"—————  —"
2130 NEXT
2140 FORK=1TO3:TB=(K-1)*10
2150 B$=LEFT$(N$(N,K),LEN(N$(N,K))-1)
2160 A$=RIGHT$(N$(N,K),1)
2170 PRINTW$"■■"TAB(TB)B$W$"■■"TAB(TB+5)A$
2180 NEXT
2190 GOTO910
2200 REM
2210 REM
2220 REM MODIFICHE DATI
2230 REM
```

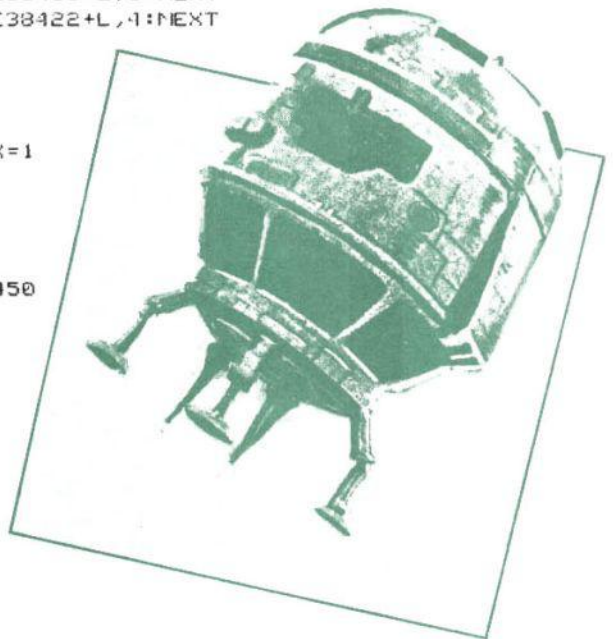
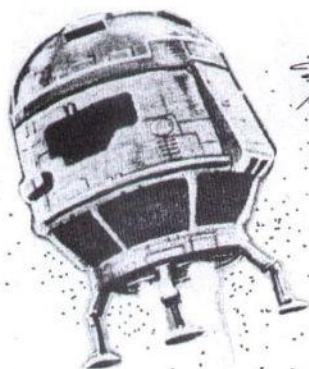
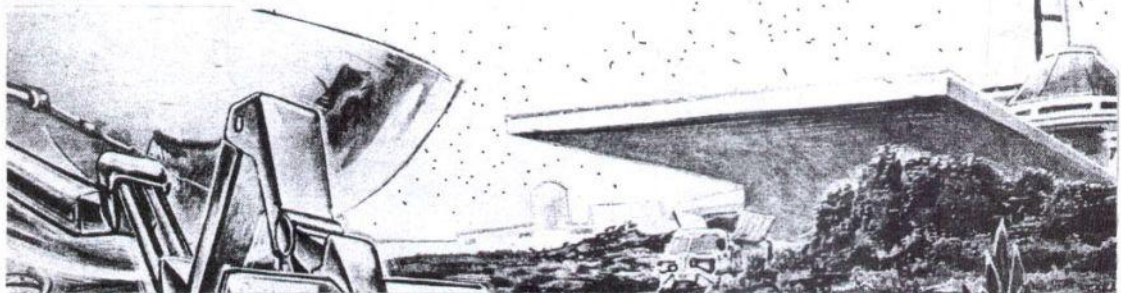
```
2240 PRINT"   INTRODUCI LE NOTE  (NOTA-OTTAVA)"
2250 GOTO880
2260 REM
2270 REM LETTURA DATI
2280 PRINT" "
2290 OPEN15,8,15,"I0":GOSUB3470
2300 INPUT"NOME DEL FILE";N$:N=0
2310 OPEN2 8,2,"0:"+N$+",S,R":GOSUB3470
2320 N=N+1
2330 FORK=1TO3:INPUT#2,N$(N,K)
2340 IFK=1THENIFN$(N,K)="FINE"THENK=4:GOTO2370
2350 NEXT
2360 GOTO2320
2370 NEXTK
2380 CLOSE2
2390 CLOSE15
2400 N1=N:N$(N,1)=""
2410 GOTO1750
2420 REM
2430 REM OUTPUT SU STAMPANTE
2440 REM
2450 PRINT"   OUTPUT SU STAMPANTE  "
2460 PRINT"  1) SOLO NOTE"
2470 PRINT" 2) SOLO DATI PER POKE NOTE"
2480 PRINT" 3) RITORNO MENU PRINCIPALE"
2490 GETA$
2500 IFA$=""THEN2490
2510 IFA$<"1"ORA$>"3"THEN2490
2520 Q=VAL(A$)
2530 ONQGOTO2550,2720,1750
2540 GOTO2490
2550 OPEN4,4:KK$="->"
2560 PRINT#4,TAB(10)"VOCE 1"TAB(15)"VOCE 2"TAB(15)"VOCE 3"
2570 PRINT#4,TAB(10)"------"TAB(15)"------"TAB(15)"------"
2580 FORK=1TON1-1
2590 K1$=N$(K,1)
2600 K2$=N$(K,2)
2610 K3$=N$(K,3)
2620 K1$=LEFT$(K1$+"      ",6):REM 6 SPAZI
2630 K2$=LEFT$(K2$+"      ",6):REM 6 SPAZI
2640 K3$=LEFT$(K3$+"      ",6):REM 6 SPAZI
2650 K$=LEFT$(STR$(K)+"   ",3):REM 3 SPAZI
2660 PRINT#4,K$KK$TAB(5)K1$TAB(15)K2$TAB(15)K3$
2670 NEXT
2680 PRINT#4," TOTALE TEMPI";N1-1
2690 PRINT#4
2700 CLOSE4
2710 GOTO2430
2720 REM
2730 REM STAPMA I VALORI DA METTERE NEI DATA
2740 REM
2750 PRINT"   ATTENDI ":KK$="->":FORI=1TON1-1
```

```
2760 FORJ=1TO3:IFN$(I,J)="↑"THENO=0:X1=0:GOTO2830
2770 O$=RIGHT$(N$(I,J),1)
2780 O=VAL(O$)
2790 N$=LEFT$(N$(I,J),LEN(N$(I,J))-1)
2800 FORX=1TO12
2810 IFS$(X)=N$THENX1=X:X=13:NEXT:GOTO2830
2820 NEXT:PRINT"■■ ERRORE ":END
2830 VL(I,J,1)=NT(O,X1,1)
2840 VL(I,J,2)=NT(O,X1,2)
2850 NEXT
2860 NEXT
2870 OPEN4,4
2880 PRINT#4,TAB(10)"VOCE 1"TAB(15)"VOCE 2"TAB(15)"VOCE 3"
2890. PRINT#4,TAB(10)"——————"TAB(15)"——————"TAB(15)"——————"
2900 FORK=1TON1-1
2910 FORJ=1TO3
2920 FORI=1TO2
2930 E$(J,I)=LEFT$(STR$(VL(K,J,I))+"      ",6)
2940 NEXT
2950 NEXT:K$=LEFT$(STR$(K)+"   ",3):REM 3 SPAZI
2960 PRINT#4,K$K$TAB(4)E$(1,1)E$(1,2)TAB(9)E$(2,1)
     E$(2,2)TAB(9)E$(3,1)E$(3,2)
2970 NEXT:PRINT#4,"■ TOTALE TEMPI"N1-1:PRINT#4,:CLOSE4:GOTO2430
2980 REM
2990 REM ASCOLTO MUSICA
3000 REM
3010 REMFORK=54296TO54360:POKEK,0:NEXT
3020 POKE54296,15
3030 POKE54277,255
3040 POKE54278,255
3050 POKE54276,33
3060 POKE54284,255
3070 POKE54285,255
3080 POKE54283,33
3090 POKE54291,255
3100 POKE54292,255
3110 POKE54290,33
3120 H1=54273
3130 H2=54280
3140 H3=54287
3150 L1=54272
3160 L2=54279
3170 L3=54286
3180 FORI=1TON1-1
3190 FORJ=1TO3:IFN$(I,J)="↑"THENO=0:X1=0:GOTO3260
3200 O$=RIGHT$(N$(I,J),1)
3210 O=VAL(O$)
3220 N$=LEFT$(N$(I,J),LEN(N$(I,J))-1)
3230 FORX=1TO12
3240 IFS$(X)=N$THENX1=X:X=13:NEXTX:GOTO3260
3250 NEXTX:PRINT"■■ ERRORE IN 23180 ":END
3260 VL(I,J,1)=NT(O,X1,1)
```

```
3270 VL(I,J,2)=NT(0,X1,2)
3280 NEXTJ
3290 NEXTI
3300 FORK=1TON1-1
3310 POKEH1,VL(K,1,1)
3320 POKEL1,VL(K,1,2)
3330 POKEH2,VL(K,2,1)
3340 POKEL2,VL(K,2,2)
3350 POKEH3,VL(K,3,1)
3360 POKEL3,VL(K,3,2)
3370 FORX=1TO99:NEXT:REM RITARDO
3380 NEXT
3390 POKEH1,0
3400 POKEH2,0
3410 POKEH3,0
3420 POKEL1,0
3430 POKEL2,0
3440 POKEL3,0
3450 GOTO1750
3460 END:REM FINE MUSICA
3470 INPUT#15,A$,B$,D$,E$
3480 IFA$="00"THENRETURN
3490 PRINT" ERRORE SU DISCO ";A$,B$
3500 REM
3510 REM SE VIENE PREMUTO F5 = INSERIRE
3520 REM
3530 IFN=N1THEN910
3540 FORY=N1+1TON+1STEP-1
3550 FORX=1TO3
3560 N$(Y,X)=N$(Y-1,X)
3570 NEXT
3580 NEXT
3590 FORX=1TO3
3600 N$(N,X)="↑"
3610 NEXT
3620 N1=N1+1
3630 PRINTW$""C$
3640 GOTO910
3650 REM
3660 REM SE VIENE PREMUTO F7 = CANCELLA
3670 REM
3680 IFN=N1THEN910
3690 FORY=NTON1-1
3700 FORX=1TO3
3710 N$(Y,X)=N$(Y+1,X)
3720 NEXT
3730 NEXT
3740 N1=N1-1
3750 PRINTW$""C$
3760 GOTO910

READY.
```

UCCIDI L'UFO

Il gioco consiste nell'abbattere 3 Ufo con i missili in dotazione (N. 10) in due minuti. Per colpire l'Ufo, il mirino deve essere sovrapposto ad esso e non distaccato.

Per ogni Ufo abbattuto si ottiene un "bonus" di 2 missili. La perdita di tutti i missili o lo scadere dei due minuti consentiti fa terminare il gioco.

Alessandro Rolla

```
100 REM *** UCCIDI L'UFO. GIOCO PER ***
110 REM ***     VIC 20 INESPANSO     ***
130 REM ***  DI  ALESSANDRO ROLLA    ***
150 :
160 PRINT"":POKE36879,59:PRINT" UCCIDI L'UFO"
170 PRINT"  ATTENDI UN ATTIMO!"
180 PRINT" ---------------------------- "
190 GOSUB1120
200 PRINT"":PRINT"USA IL JOYSTIK PER"
210 PRINT"MUOVERE IL MIRINO E"
220 PRINT"COLPIRE UN UFO CON"
230 PRINT"UNA BOMBA."
240 PRINT"HAI 10 BOMBE."
250 PRINT"DEVI ABBATTERE 3 UFO."
260 PRINT"AD OGNI CENTRO VINCI":PRINT"DUE MISSILI."
270 PRINT"------------------------------":REM SHIFT+C
280 PRINT"PREMI QUALSIASI TASTO"
290 GETA$:IFA$=""THEN290
```

```
300 PRINT"HAI DUE MINUTI DI ":PRINT"TEMPO PER IL GIOCO."
310 PRINT"SE PERDI TUTTI I     "
320 PRINT"MISSILI,MUORI!"
330 PRINT"     BUONA FORTUNA  "
340 PRINT"                   "
350 GETA$:IFA$=""THEN350
360 PRINT""
370 POKE36869,255:POKE36879,8:A=37137:B=37152:S1=36877:V2=9:Z=3
380 TI$="000000"
390 FORL=0TOV2:POKE7680+L,36:POKE38400+L,3:NEXT
400 FORL=0TO21:POKE7702+L,42:POKE38422+L,4:NEXT
410 POKE37154,127:POKE36878,15
420 X=1:DX=1:M=1:G=67
430 POKE7800+X,34:POKE38520+X,7
440 POKE8000+M1,32:GOTO550
450 D=INT(RND(1)*4)+1:IFD=1THENDX=1
460 IFD=2THENDX=-1
470 IFD=3THENDX=22
480 IFD=4THENDX=-22
490 X1=X:X=X+DX
500 IFX=<-54ORX=>98THENX=X1:GOTO450
510 POKE7800+X1,32
520 PRINT"TIME:"TI$
530 IFVAL(TI$)=>200THEN1080
540 GOTO430
550 POKE8000+M,35:POKE38720+M,5
560 AB=PEEK(A)+PEEK(B)
570 M1=M:IFAB=341THEN640
580 IFAB=369THENM=M-22
590 IFAB=365THENM=M+22
600 IFAB=357THENM=M-1
610 IFAB=245THENM=M+1
620 IFM=<-277THENM=M1
630 GOTO450
640 IF7800+X=8000+M THENGOSUB1200:POKE7823+X,41
650 POKE38497+X,7:POKE38499+X,7:POKE38541+X,7:POKE38543+X,7
660 K=M:POKE7800+X,34:FORI=1TO15
670 POKE8000+K+330,G
680 POKE38720+K+330,3
690 FORJ=1TO30:NEXT
700 IFI=5THENG=37
710 IFI=10THENG=38
720 POKES1,220+I*2
730 T=K:K=K-22
740 POKE8000+T+330,32
750 PRINT""TI$
760 NEXT
770 POKE7680+V2,32:V2=V2-1
780 G=36:POKES1,0:IFPEEK(8000+K+330)=34THEN810
790 IFV2=-1THEN970
800 GOTO450
810 POKE7800+X,39:POKE38520+X,2
```

```
820 T1$=TI$
830 POKES1,220
840 FORL=15TO0STEP-1
850 POKE36878,L
860 FORM=1TO300:NEXTM,L
870 POKES1,0:POKES1+1,15
880 POKE7777+X,32:POKE7779+X,32:POKE7821+X,32
890 POKE7823+X,32:POKE7800+X,32
900 Z=Z-1:IFZ=0THEN1010
910 FORP=1TO10:PRINT"          BONUS"
920 FORT=1TO200:NEXT:POKES1-1,220:PRINT"
930 FORT=1TO200:NEXT:POKES1-1,0:NEXT
940 V2=V2+2
950 TI$=T1$
960 GOTO390
970 PRINT"MISSILI ESAURITI"
980 PRINT"****************"
990 PRINT"ANCORA       [S/N]"
1000 GOTO1040
1010 PRINT"COMPLIMENTI,HAI VINTO"
1020 PRINT"*********************"
1030 PRINT"ANCORA       [S/N]"
1040 GETA$:IFA$=""THEN1040
1050 IFA$="S"THENPRINT"":RUN370
1060 IFA$="N"THENEND
1070 GOTO1040
1080 PRINT"TEMPO SCADUTO"
1090 PRINT"*************"
1100 PRINT"ANCORA       [S/N]"
1110 GOTO1040
1120 FORI=0TO511:POKE7168+I,PEEK(32768+I):NEXT
1130 FORI=0TO71:READA:POKE7440+I,A:NEXT:RETURN
1140 DATA 24,60,231,60,60,24,36,36,0,8,8
1150 DATA 8,119,8,8,8,60,126,255,255,126,60,24,60
1160 DATA 60,126,126,60,24,60,0,0,56,124
1170 DATA 56,16,56,0,0,0,128,37,16,42,0,40,36,18
1180 DATA 1,2,4,8,16,32,64,128,128,64
1190 DATA 32,16,8,4,2,1,255,255,0,0,0,0,0,0
1200 POKE7777+X,41:POKE7779+X,40:POKE7821+X,40:RETURN

READY.
```

DUE SEMPLICI ...
ROUTINE MATEMATICHE

Esercizi di digitazione utili non solo per prender confidenza con la tastiera ma anche per "utilizzare" il calcolatore.

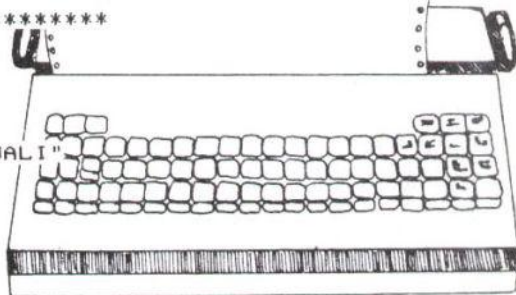
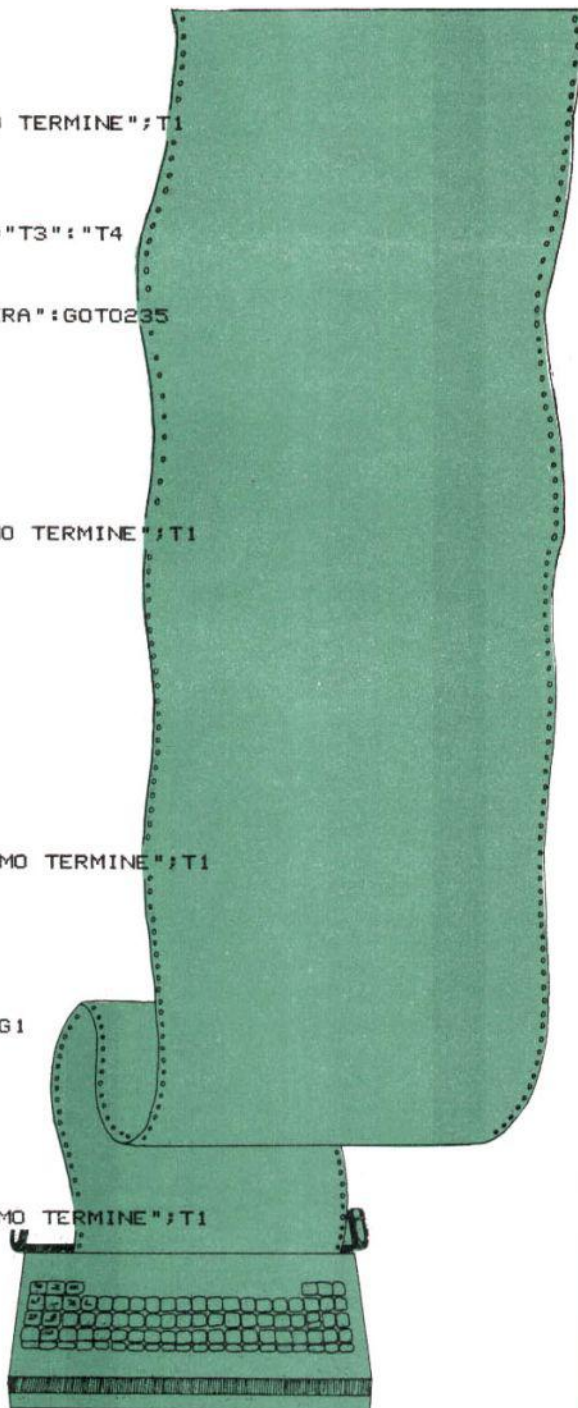
Il programma DIVISORE permette di conoscere in un tempo relativamente breve (6'30" per il numero 999999999) tutti i valori per cui un numero X è divisibile.

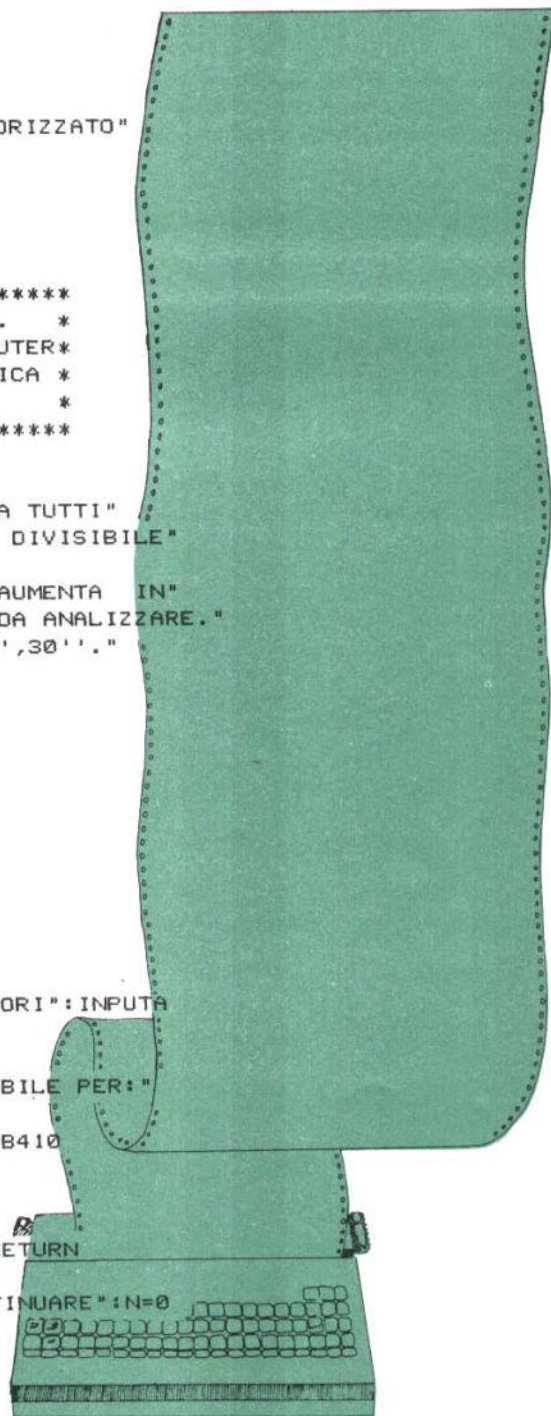
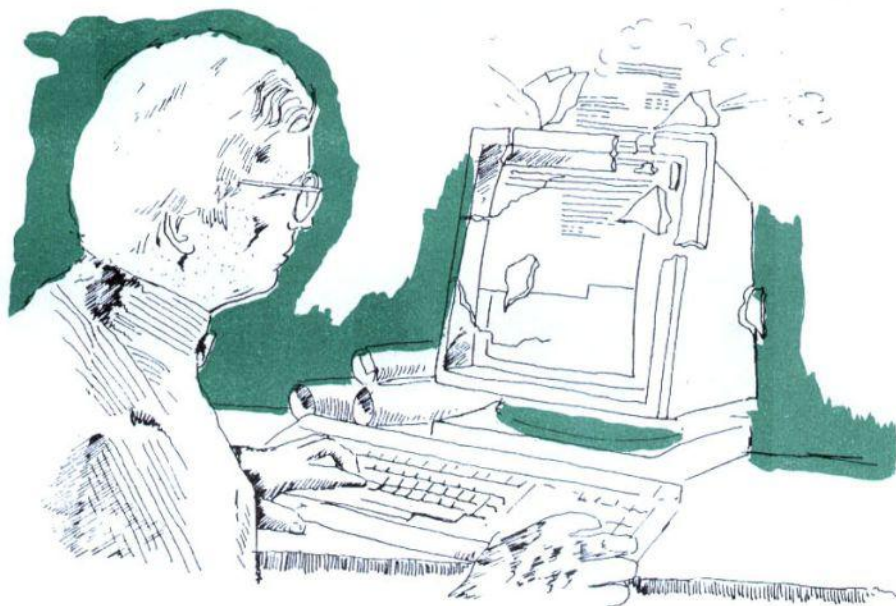
La visualizzazione su due colonne permette di render visibili i divisori che, moltiplicati tra loro, riproducono il numero in esame.

Il listato PROPORZIONI consente, appunto, di risolverle e di verificarle anche nei casi in cui si debbano calcolare le percentuali.

I due programmi sono talmente semplici che non necessitano di commenti. Sono pubblicati in forma "povera". Al lettore il compito di sofisticarli da un punto di vista estetico e (perché no?) di contenuti.

Edie Meglio, Giuliano Rossi

```
1 REM  ***************************************
10 REM *      L'ANGOLO DEI PRINCIPIANTI:        *
11 REM *PROPORZIONI (PER QUALSIASI COMPUTER)*
12 REM *   SEMPLICE LISTATO DI MATEMATICA      *
13 REM *            DI EDIE MIGLIO              *
14 REM ***************************************
15 :
16 PRINT
19 PRINT"1.VERIFICA PROPORZIONE"
21 PRINT"2.CALCOLO <X> ESTREMA"
22 PRINT"3.PROPORZIONE CON DUE <X> UGUALI"
23 PRINT"4.CALCOLO <X> MEDIA"
24 PRINT"5.USCITA":PRINT
25 INPUT"SCELTA";SC
26 IFSC<1ORSC>5 THEN25
30 ON SC GOTO 100,300,500,700,900
```

```
100 REM ******************
101 REM *     VERIFICA     *
102 REM * PROPORZIONALITA'*
103 REM ******************
110 PRINT"A : B = C :D":INPUT"PRIMO TERMINE";T1
111  INPUT"SECONDO TERMINE";T2
112  INPUT"TERZO TERMINE";T3
113  INPUT"QUARTO TERMINE";T4
115 PRINT"LA PROPORZIONE "T1":"T2"="T3":"T4
120 K1=T1/T2
125 K2=T3/T4
230 PRINT"E' ";:IFK1=K2THENPRINT"VERA":GOTO235
231 PRINT"FALSA"
235 PRINT"PREMI UN TASTO"
245 GETW$:IFW$=""THEN245
250 GOTO16
300 REM *************
301 REM * UNA SOLA INCOGNITA
302 REM *************
310 PRINT"A : B = C : X":INPUT"PRIMO TERMINE";T1
315 INPUT"SECONDO TERMINE";T2
320 INPUT"TERZO TERMINE";T3
321 PRINT"QUARTO TERMINE? X"
330 J1=T2*T3/T1
335 PRINT"LA <X> E':"J1
340 PRINT"PREMI UN TASTO"
345 GETW$:IFW$=""THEN345
346 GOTO16
500 REM ****************
501 REM * DUE <X> UGUALI*
502 REM ****************
505 PRINT"A : X = X : B":INPUT"█PRIMO TERMINE";T1
510 PRINT"SECONDO TERMINE= X"
515 PRINT"TERZO TERMINE= X"
520 INPUT"QUARTO TERMINE";T2
530 H1=T1*T2
535 G1=SQR(H1)
540 PRINT"IL TERMINE INCOGNITO E':"G1
550 PRINT"PREMI UN TASTO"
555 GETW$:IFW$=""THEN555
556 GOTO16
700 REM ********************
701 REM * CALCOLO <X> MEDIA *
702 REM ********************
705 PRINT"A : X = B : C█":INPUT"PRIMO TERMINE";T1
710 PRINT"SECONDO TERMINE= X"
720 INPUT"TERZO TERMINE";T2
725 INPUT"QUARTO TERMINE";T3
730 W1=T1*T3/T2
750 PRINT"L'INCOGNITA E':"W1
760 PRINT"PREMI UN TASTO"
765 GETW$:IFW$=""THEN765
```

```
766 GOTO 16
900 REM ********
901 REM * FINE *
902 REM ********
905 PRINT"IL PROGRAMMA E' ANCORA MEMORIZZATO"
910 END
READY.
```

```
70 REM **********************************
75 REM *   L'ANGOLO DEL PRINCIPIANTE.   *
80 REM *DIVISORE: PER QUALSIASI COMPUTER*
85 REM * SEMPLICE LISTATO DI MATEMATICA *
86 REM *        DI GIULIANO ROSSI       *
90 REM **********************************
100 :
130 :
150 PRINT"QUESTO PROGRAMMA VISUALIZZA TUTTI"
160 PRINT"I NUMERI INTERI PER CUI E' DIVISIBILE"
180 PRINT"IL NUMERO IMPOSTATO."
185 PRINT"IL TEMPO DI  ELABORAZIONE AUMENTA  IN"
190 PRINT"PROPORZIONE CON IL NUMERO DA ANALIZZARE."
200 PRINT"ES.: PER 999999999 E' DI 6',30''."
220 :
240 REM *SCELTA VISUALIZZAZIONE*
250 PRINT"┌─────────────┐"
260 PRINT"|  USCITA     |"
270 PRINT"├──┬──────────┤"
280 PRINT"|3 |VIDEO     |"
290 PRINT"|4 |STAMPANTE |"
300 PRINT"└──┴──────────┘"
310 INPUT"SCELTA (3/4) ";A$
330 IFA$="3"ORA$="4"THEN350
340 GOTO310
350 U=VAL(A$)
353 PRINT"DI CHE NUMERO VUOI I DIVISORI":INPUTA
355 OPEN1,U
359 REM *CALCOLO DIVISORI*
360 PRINT#1,"IL NUMERO";A;"E' DIVISIBILE PER:"
370 B=INT(SQR(A)):N=0
380 FORC=1TOB:IFA/C=INT(A/C)THENGOSUB410
390 NEXT:GOTO470
400 REM *VISUALIZZAZIONE DIVISORI*
410 IFC=A/CTHENPRINT#1,C:RETURN
420 PRINT#1,C,A/C:N=N+1:IFN<>20THENRETURN
430 IFU<>3THENRETURN
440 PRINT#1,"PREMI UN TASTO PER CONTINUARE":N=0
450 GETA$:IFA$=""THEN450
460 RETURN
470 PRINT#1:CLOSE1:GOTO250
READY.
```

SCROLLING TOOL

Un tool di scrolling nelle quattro direzioni di schermo per il Commodore 64

Alcuni di voi, leggendo il titolo, si saranno forse chiesti se lo scrolling è un gruppo musicale che ricalca le orme dei Rolling Stones oppure un sistema americano per scrollarsi più velocemente le pulci di dosso.

Niente di tutto ciò: lo scrolling è.... lasciamo a parte un attimo le spiegazioni e facciamo una piccola prova direttamente sul computer:

- premiamo contemporaneamente i tasti SHIFT-CLR HOME.
- premiamo RETURN 5 volte di seguito
- scriviamo: STO IMPARANDO LE TECNICHE DI SCROLLING
- teniamo premuto il tasto CURSOR-DOWN (il secondo in basso a destra)

Noteremo che il messaggio prima digitato si sposta velocemente verso l'alto, fino a scomparire. Che cos'è successo? Il computer ha applicato allo schermo una routine particolare, detta di scrolling, che ha il compito di spostare la scritta sempre più in alto tanto da farla uscire al di fuori dello schermo. Avviene la stessa cosa eseguendo il comando LIST, o con una serie di istruzioni PRINT, nel caso in cui si superi il numero delle venticinque righe disponibili.

Possiamo così concludere che lo scrolling è un programma (Basic o L.M.) che sposta in una direzione (nel caso esaminato verso l'alto) tutto ciò che è presente sullo schermo allo scopo di creare nuovo spazio.

Lo scrolling tool pubblicato serve a fornire al Commodore 64, che ne è carente, la possibilità di scroll (è l'abbreviazione che si usa normalmente) in tutte le direzioni. A che cosa serve, in definitiva, il programma? Agli impazienti consigliamo di digitare, subito dopo il programma, anche i brevi listati dimostrativi.

Il programma contiene, nella fase di lettura dei DATA, controlli particolari di checksum (somma dei DATA stessi) utili per rintracciare eventuali errori e fornirne il numero del blocco in cui sono stati digitati. È sempre consigliabile, ad ogni buon conto, registrare il programma subito dopo averlo digitato. Una maggior prudenza non guasta mai ed evita spiacevoli sorprese.

Come si usa

Partito il programma (RUN), seguirà una brevissima dimostrazione di controllo dopo la quale sarà pronto per l'uso. I comandi sono semplicissimi: si digita il tasto di freccia a sinistra (primo tasto in alto a sinistra) e subito dopo la direzione o le direzioni in cui si vuole eseguire lo scroll. I caratteri che è possibile digitare dopo la freccetta hanno la sola limitazione del numero dato dalla consueta lunghezza delle righe Basic: non più di 80 caratteri in tota-

le.

I caratteri indicanti le quattro direzioni derivano dalle iniziali delle parole inglesi:
U (up) = alto
D (down) = basso
R (right) = destra
L (left) = sinistra

Queste sono però le stesse lettere riportate tra la riga 100 e la riga 130. Non è che per caso...?
Esatto: c'è la possibilità di ridefinire i comandi per le direzioni cambiando semplicemente quelle lettere con un carattere qualsiasi!!

Nella prima versione c'era il set di istruzioni in italiano (A, B, D, S), ma è una sequenza che sconsigliamo dato che è facile comporre la parola basic ABS con conseguente ?SYNTAX ERROR. Egual messaggio di errore sarà generato nel caso che le lettere poste dopo la freccia non corrispondano parzialmente o totalmente a quelle impostate.
Per rendere i programmi compatibili, qualunque sia il carattere-codice scelto, è possibile utilizzare le SYS riportate all'inizio di ogni routine (vedi righe 290-350). I più esperti potranno notare, dall'indirizzo di partenza, che la routine di up scroll è proprio quella del Sistema Operativo del C-64.

Un ultimo avvertimento: a causa dell'aggiunta dei nuovi comandi premendo il tasto Return dopo il solo carattere due punti (:) non si avrà più "READY." ma un messaggio di errore. Accadrà la stessa cosa tentando di porre il solo carattere di doppio punto in un riga Basic.

Come funziona lo scroll

Come abbiamo già detto, lo scroll è un programma che sposta tutti i caratteri presenti sullo schermo verso una direzione. Il modo in cui agisce è più semplice di quanto possa sembrare. Iniziamo col digitare il programma della figura 1, scriviamo sullo schermo caratteri a caso, possibilmente di colori diversi, e facciamo partire il programma.
Vedremo, a partire dall'alto, le righe che verrano copiate in quelle immediatamente superiori e avremo un momentaneo effetto di sdoppiamento.

In seguito la riga inferiore si riempirà di spazi vuoti e subito dopo apparirà il messaggio READY.

Ora proviamo a scrivere la nuova riga:
50 GOTO 20

Otterremo lo stesso effetto dell'esperimento fatto all'inizio premendo il tasto CURSOR DOWN.

Analizzando il programma nella riga 10 troviamo la definizione delle variabili usate:

SC = inizio della memoria di schermo
CO = inizio della memoria colore
ROW = numero di locazioni per riga di schermo.

Nella riga 20 il ciclo FOR NEXT parte dalla riga di schermo 0 e si conclude con la fine della riga di schermo 23. A questo punto abbiamo la "copia" delle locazioni di memoria di schermo che per la memoria colore.

La riga 25 provvede a chiudere il ciclo; mentre con le istruzioni della riga 30 viene riempita di spazi l'ultima riga di schermo.

Possiamo così affermare che per effettuare uno scrolling si parte dalla prima locazione di memoria nella direzione verso cui si vuole 'scrollare' e si arriva alla locazione di fine schermo meno uno spostamento, cioè 40 caratteri. In questo ciclo si sposta nella locazione attuale il contenuto della locazione (±1) spostamento, infine si azzera la riga entrante per creare nuovo spazio.

Tale procedura è confermata dal programma di down scroll della figura 2. Per i programmi delle figure 2 e 3 il concetto non cambia. Le uniche differenze consistono nel fatto che lo spostamento è rappresentato da CL=colonna, e viene azzerata prima la colonna entrante, quella a sinistra per lo spostamento a sinistra e quella a destra per lo spostamento a destra, mentre alla fine viene cancellato l'ultimo carattere entrante che prima non era presente sullo schermo.

Le routines L.M. di scrolling tool fanno tutto ciò che abbiamo descritto ma, ovviamente, in modo molto più veloce.

I programmi dimostrativi

Qualche parola si rende necessaria per utilizzare i due programmi dimostrativi, che vanno usati con lo scrolling tool nella versione pubblicata (senza cambiare le lettere che indicano le direzioni).

La prima dimostrazione è una corsa automobilistica a tempo con scrolling verso il basso, quindi molto più realistica delle altre versioni cui siete abituati (avete mai visto una corsa in retromarcia?).

In questo programma le righe da 90 a 96 servono a inizializzare il programma, quelle da 100 a 130 a generare la pista, e dalla riga 140 fino alla 200 per controllare se si esce fuori pista o per finire il gioco.

L'altro brevissimo programma serve per fare scorrere delle scritte sullo schermo. Le opzioni "singola" e "multipla" servono rispettivamente per fare scorrere la scritta contemporaneamente su una o su tutte le righe dello schermo.

Fabio Sorgato

```
10 REM **  SCROLLING TOOL  **
20 REM ** BY FABIO SORGATO **
30 REM **                  **
40 REM
100 A1$="U":REM UP (ALTO)
110 A2$="D":REM DOWN (BASSO)
120 A3$="R":REM RIGHT (DESTRA)
```

```
130 A4$="L":REM LEFT (SINISTRA)
140 REM
150 READN,S
160 FORK=49152TO49492:READA:R=R+1:CK=CK+A:IFR<>N AND A<256THEN200
170 B=B+1:IFCK<>S OR A>255THENPRINT"ERRORE DI DATI NEL BLOCCO #"B:END
180 IFB=5THEN220
190 CK=0:R=0:READN,S
200 POKEK,A:NEXT
210 REM ASSEGNA CODICE COMANDI
220 POKE49446,ASC(A1$)
230 POKE49456,ASC(A2$)
240 POKE49466,ASC(A3$)
250 POKE49476,ASC(A4$)
260 SYS49421:REM START
270 A$="[illegible]"
280 PRINTA$;"PROVA SCROLLING":FORT=1TO800:NEXT
290 PRINTA$;"UP SCROLL":A=59626:GOSUB350
300 PRINTA$;"DOWN SCROLL":A=49180:GOSUB350
310 PRINTA$;"RIGHT SCROLL":A=49329:GOSUB350
320 PRINTA$;"LEFT SCROLL":A=49249:GOSUB350
330 PRINT"                  SCROLLING-TOOL ATTIVATO"
331 PRINT"ESEMPIO:←DDDDDDDLLLLLLLRRRRRDDDDD"
332 ←DDDDDDDLLLLLLLRRRRRDDDDD
340 END
350 FORK=1TO500:NEXT:FORK=1TO30:SYSA:NEXT:RETURN
1000 :
1010 :REM BLOCCO #1
1020 :
1030 :DATA 28,3520:REM N' DATI,CHECKSUM
1040 DATA 169,0,133,2,168,138
1050 DATA 145,251,152,24,105,40
1060 DATA 144,2,230,252,164,2
1070 DATA 192,24,240,5,168,230
1080 DATA 2,208,234,96
1090 :
1100 REM BLOCCO #2
1110 :
1120 REM *******************
1130 REM **   DOWN SCROLL   **
1140 REM ** $ C01C,# 49180 **
1150 REM *******************
1160 :
1170 DATA 69,11932:REM N' DATI,CHECKSUM
1180 DATA 169,232,133,251,169,192,133
1190 DATA 253,162,6,160,255,134,252
1200 DATA 134,254,177,253,145,251
1210 DATA 138,24,105,212,133,252
1220 DATA 133,254,177,253,145,251
1230 DATA 224,3,208,4,192,64
1240 DATA 240,18,136,208,225,177
1250 DATA 253,145,251,134,254,134
1260 DATA 252,177,253,145,251,202
```

```
1270 DATA 208,208,162,40,169,32
1280 DATA 157,255,3,202,208,250
1290 DATA 96
1300 :
1310 REM BLOCCO #3
1320 :
1330 REM ********************
1340 REM **    LEFT SCROLL   **
1350 REM ** $ C061,# 49249 **
1360 REM ********************
1370 :
1380 DATA 80,12196:REM N' DATI,CHECKSUM
1390 DATA 169,0,133,251,169
1400 DATA 4,133,252,162,32,32
1410 DATA 0,192,169,216,133,252
1420 DATA 174,33,208,32,0,192
1430 DATA 162,0,134,251,232,134
1440 DATA 253,162,4,160,0,134
1450 DATA 252,134,254,177,253,145
1460 DATA 251,138,24,105,212,133
1470 DATA 252,133,254,177,253,145
1480 DATA 251,224,7,208,4,192
1490 DATA 232,240,6,200,208,225
1500 DATA 232,208,220,169,32,141
1510 DATA 231,7,174,33,208,142
1520 DATA 231,219,96
1530 :
1540 REM BLOCCO #4
1550 :
1560 REM ********************
1570 REM **  RIGHT SCROLL   **
1580 REM ** $ C0B1,# 49329 **
1590 REM ********************
1600 :
1610 DATA 32,14343:REM N' DATI,CHECKSUM
1620 DATA 169,39,133
1630 DATA 251,169,4,133,252,162
1640 DATA 32,32,0,192,169,216
1650 DATA 133,252,174,33,208,32
1660 DATA 0,192,162,232,134,251
1670 DATA 202,134,253,162,6,160
1680 DATA 255,134,252,134,254,177
1690 DATA 253,145,251,138,24,105
1700 DATA 212,133,252,133,254,177
1710 DATA 253,145,251,224,3,208
1720 DATA 4,192,25,240,16,136
1730 DATA 208,225,177,253,145,251
1740 DATA 134,252,134,254,177,253
1750 DATA 145,251,202,208,208,169
1760 DATA 32,141,0,4,173,33
1770 DATA 208,141,0,216,96
1780 :
```

```
1790 REM ********************
1800 REM **    UP SCROLL    **
1810 REM ** $ E8EA,# 59626 **
1820 REM ********************
1830 :
1840 REM BLOCCO #5
1850 :
1860 REM ***************************
1870 REM **   RICONOSCE COMANDI    **
1880 REM ** E START $C10D,# 49421 **
1890 REM ***************************
1900 :
1910 DATA 71,7459:REM N' DATI,CHECKSUM
1920 DATA 169,24,141,8,3,169,193,141
1930 DATA 9,3,96,32,115,0,201,95
1940 DATA 240,3,76,231,167,32,115,0
1950 DATA 201,65,208,6,32,234,232,76
1960 DATA 34,193,201,66,208,6,32,28
1970 DATA 192,76,34,193,201,68,208,6
1980 DATA 32,177,192,76,34,193,201,83
1990 DATA 208,6,32,97,192,76,34,193
2000 DATA 32,121,0,76,174,167,0
READY.
```

```
1 REM *** ESEMPIO SCROLL IN BASIC ***
2 :
10 SC=1024:CO=55296:ROW=40
20 FOR K=0*ROW TO 24*ROW-1
21 POKE SC+K,PEEK(SC+K+ROW):POKE CO+K,PEEK(CO+K+ROW)
25 NEXT
30 FOR K=24*ROW TO 25*ROW -1:POKE K+SC,32:POKE CO+K,6:NEXT
READY.
```

```
2 REM *** SCROLL VERTICALE IN BASIC ***
3 :
10 SC=1024:CO=55296:ROW=40
20 FOR K=24*ROW-1 TO 0 STEP-1:POKE SC+ROW+K,PEEK(SC+K)
30 POKE CO+ROW+K,PEEK(CO+K):NEXT
40 FOR K=ROW*0 TO 1*ROW -1:POKE K+SC,32:POKE K+CO,6:NEXT
READY.
```

```
3 REM *** SCROLL SINISTRO IN BASIC ***
4 :
10 SC=1024:CO=55296:ROW=40:CL=1
20 FOR K=ROW*0 TO ROW*24 STEP ROW:POKE K+SC,32:POKE K+CO,6:NEXT
30 FORK=ROW*0 TO ROW*25-1
35 POKE K+SC,PEEK(K+SC+CL):POKE K+CO,PEEK(K+CO+CL):NEXT
40 POKE SC+25*ROW-1,32:POKE CO+25*ROW-1,6
READY.
```

```
4 REM *** SCROLL DESTRO IN BASIC ***
5 :
10 SC=1024:CO=55296:ROW=40:CL=1
20 FOR K=ROW*1-1 TO ROW*25-1 STEP ROW
25 POKE K+SC,32:POKE K+CO,6:NEXT
30 FOR K=ROW*25-1 TO ROW*0 STEP-1
35 POKE K+SC+CL,PEEK(K+SC)
40 POKE K+CO+CL,PEEK(K+CO):NEXT
50 POKE SC+ROW*0,32:POKE CO+ROW*0,6
READY.
```

```
100 REM *** GIOCO APPLICATIVO DELLO SCROLLING TOOL ***
110 REM ***                                        ***
115 REM *** USARE I TASTI VIRGOLA (,) E PUNTO (.)  ***
120 REM
130 PRINT"■":POKE650,128
140 INPUT"LARGHEZZA STRADA (MAX 20)";L:IFL>20ORL<2THEN130
150 X=(40-L)/2-1:D=20:PRINT"■":L=L+1:B=170:C=165
160 FORT=0TO20:PRINT"■"TAB(X)CHR$(B);CHR$(B);
170 PRINTTAB(X+L+1);CHR$(C);CHR$(C):←D:NEXT
180 TI$="000000"
130 A=INT(RND(1)*3):X1=X:R=0:Q=0:B1=B
200 IFA=0THENA$=CHR$(206):B$=A$:IFB1=BTHENX=X+1
210 IFA=1THENA$=CHR$(205):B$=A$:Q=1:IFB1=BTHENX=X-1
220 IFA=2THENA$=CHR$(B):B$=CHR$(C):R=1
230 IFX<1ORX+L+R+Q>38THENX=X1:A$=CHR$(B):B$=CHR$(C):R=1
240 PRINT"■";TAB(X+Q);A$;A$;TAB(L+X+R+Q);B$;B$
250 GETK$:IFK$=""THEN300
260 IFK$=","THEND=D-1:IFD<1THEND=1
270 IFK$="."THEND=D+1:IFD>37THEND=37
280 K=1504+D
290 IFPEEK(K)=32ORPEEK(K)=81THENPOKEK1,32:←D:POKEK,81:K1=K:GOTO190
300 K=1504-40+D:IFPEEK(K)=32THENPOKEK1,32:←D:POKEK+40,81:K1=K+40:GOTO190
310 T$=TI$:POKEK1,42:FORK=1TO1000:NEXT
320 PRINT"■HAI GIOCATO PER UN TEMPO DI ";T$
330 FORK=1TO2000:NEXT:POKE198,0:RUN
READY.
```

```
1 REM *** SCROLLING TOOL DEMO ***
2 REM
10 INPUT"DIGITA UNA FRASE";A$
12 REM LA PROSSIMA STRINGA CONTIENE 40 SPAZI
15 A$=A$+"                                        ":REM
20 INPUT"■S■INGOLA O ■M■ULTIPLA";B$
25 IFB$<>"S"ANDB$<>"M"THEN20
30 PRINT"■":IFB$="M"THEN60
40 PRINT"■■■■■■■■■■■■■■■"TAB(40);
50 FORK=1TOLEN(A$):PRINT"■"MID$(A$,K,1);
55 ←L:FORT=1TO70:NEXT:NEXT:GOTO50
60 FORK=1TOLEN(A$):PRINT"■"TAB(40);
70 FORT=1TO23:PRINT"■■"MID$(A$,K,1);:NEXT:←L:NEXT:GOTO60
READY.
```

CARICAMENTO RAPIDO DI ROUTINE IN L.M.

Come ridurre al minimo i tempi di caricamento di programmi in linguaggio macchina generati da listati Basic

Allo scopo di effettuare una pratica applicazione dell'argomento proposto, viene qui di seguito illustrata la procedura da seguire per memorizzare le routine grafiche di D. Toma. Vi sarete accorti, infatti, che le favolose routines apparse sul numero 14 di C.C.C. hanno due piccoli difetti: non solo è necessario parecchio tempo per caricare il programma Basic che le genera e le alloca in memoria, ma la lettura dei dati richiede altro tempo prezioso (e il tempo, si sa, è denaro).

Il breve listato pubblicato serve a ridurre il primo e ad annullare il secondo difetto, dato che consiste in una routine l.m. che sposta i dati delle routines grafiche nella zona bassa di memoria e aggiunge automaticamente la riga Basic:

10 sys 2063: new

Viene inoltre addizionata una nuova routine l.m. che riporterà i dati, dopo il caricamento, nella posizione originale ed eseguirà le due SYS di partenza indispensabili per "inizializzare" il programma. L'uso del programma è molto semplice: dapprima si carica e si "lancia" il programma Basic che contiene le routines grafiche. Si carica quindi il programmino pubblicato in queste pagine e lo si fa partire.

La risposta sarà:

?SYNTAX ERROR IN 120

Non spaventatevi: non avete commesso alcun errore.

Il fenomeno si spiega col fatto che al ritorno dalla routine l.m., l'interprete Basic non trova ciò che si aspettava, ma alcuni codici in Linguaggio Macchina senza senso e genera, di conseguenza, un messaggio di errore.

Eseguite un LIST per controllare che tutto sia in ordine (10 SYS2063:NEW) e poi registrate, col nome che più vi aggrada, il programma generato in memoria che non occuperà più 9 Kilobytes, ma i reali 2 K (meno di un quarto del tempo per caricare il programma).

Al momento del RUN, inoltre, la risposta sarà immediata.

Come già detto all'inizio, i più bravi, disassemblando la routine proposta, troveranno di sicuro utili indicazioni per applicarla in altri casi analoghi, come quelli in cui non solo è necessario "caricare" un programma L.M. nelle giuste locazioni RAM, ma anche impartire comandi SYS per inizializzarle.

Fabio Sorgato

```
5 :REM ********************************
10 REM ** TRASFERIMENTO RAPIDO DI     **
20 REM ** ROUTINES IN LINGUAGGIO M.  **
55 REM ********************************
60 REM **       BY  FABIO SORGATO     **
80 REM **    ESEMPIO: TRASFERIMENTO   **
85 REM **DELLE ROUTINES GRAFICHE II  **
90 REM ********************************
120 FORK=24576TO24685:READA:POKEK,A:CK=CK+A:NEXT
110 IFCK<>15718THENPRINT"ERRORE NEI DATI":END
120 SYS24576:END
1000 DATA162,58,189,53,96,157.0,8
1010 DATA202,208,247,160,0,132,251,169
1020 DATA192,133,252,169,64,133,253,169
1030 DATA8 133,254,177,251,145,253,200
1040 DATA208,249,230,252,230,254,165
1050 DATA254,201,17,208.239,169,72,133
1060 DATA45,169,16,133,46,96,0,13,8
1070 DATA10,0,158,50,48,54,51,58
1080 DATA162.0,0,0,169,64,133,251
1090 DATA169,8,133,252,169,0,133,253
1100 DATA169;192.133,254,160.0,177,251
1110 DATA145,253,200,208,249,230,252
1120 DATA230,254,165,252,201,17,208,239
1130 DATA32.58.199,32,60,194,96
READY.
```

CERCASI

La redazione di Commodore Computer Club vuole potenziarsi e ricerca collaboratori part-time e insegnanti di discipline scientifiche e tecniche preferibilmente residenti nell'area di Milano

Ai collaboratori che stiamo ricercando verrà richiesto di collaborare alle varie iniziative della casa editrice con articoli, libri, raccolte di programmi e l'italianizzazione di software, di cui abbiamo i diritti d'autore, orientati alla didattica per le scuole medie e superiori.
I prescelti, pertanto dovranno possedere un sistema completo di Vic 20 oppure Commodore 64 e sapere programmare sia in basic che in linguaggio macchina. La conoscenza dell'inglese e di altri personal computer è un titolo preferenziale.

Compensi

Tutti i lavori svolti su incarico della redazione verranno sempre compensati in base ai miglior standard di mercato.

Primo contatto

Per incontrarci telefonate allo 02/8467348 chiedendo della signorina Piera

NUMERI PRIMI

Numeri primi di Fermat. Questo programma stampa una serie di numeri primi introducendo gli estremi inferiore e superiore. L'algoritmo adottato non si basa sulla fattorizzazione, ma su un teorema formulato per la prima volta da Pierre de Fermat nel 1640.

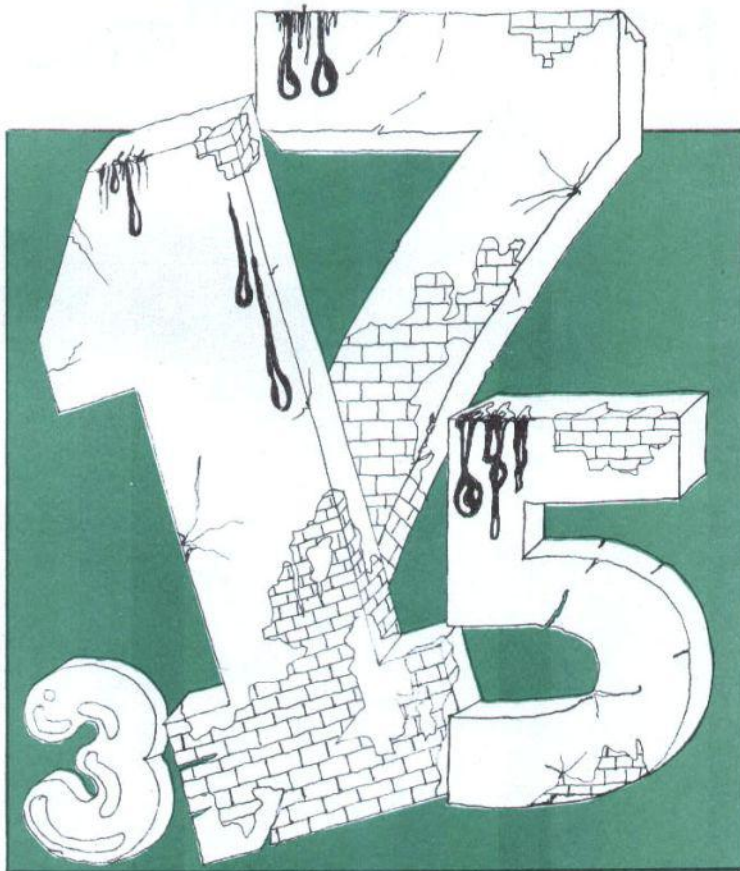
Questo stabilisce che se N è un numero primo e B è un intero qualsiasi, allora B elevato alla potenza "n" - B è un multiplo di N; tralascio volutamente ulteriori dettagli matematici. Per problemi di arrotondamento non si possono superare valori maggiori di 2^{16}: 65536.

Fattori primi, un programma realizzato parzialmente in L.M., scompone in fattori primi numeri fino a dieci cifre con il sistema delle divisioni per tentativi.

Uno dei problemi che ha affascinato e impegnato i matematici di tutti i tempi è senz'altro l'individuazione dei numeri primi.

Fino a pochi secoli fa dimostrare la primalità di un numero di una decina di cifre era un lavoro non da poco dato che allora l'unico sistema per fare calcoli era del tipo... inchiostro, sudore e... carta.

Per citare un esempio pratico bisogna aspettare il 1772 quando ad opera del matematico Leonardo Eulero, si dimostrò che due elevato alla 31ma potenza (meno uno) (=2147483647) era primo e considerando che all'epoca Eulero era quasi cieco, l'impresa ha davvero dell'incredibile.

Ai giorni nostri, volendo cimentarsi in problemi del genere, non è più necessario possedere la genialità di Eulero e la pazienza di un certosino, ma è sufficiente avere a portata di dita un piccolo calcolatore ed è abbastanza semplice scrivere un programma in Basic per testare la primalità di un numero.

Il numero più ovvio e semplice consiste nel dividere il numero in esame N per i numeri della successione 2, 3, 4... N; se qualcuna delle divisioni non dà resto allora è trovato un fattore di N, in caso contrario N sarà primo.

È possibile migliorare l'efficienza dell'algoritmo limitando le divisioni alla radice di N ed eliminando i divisori pari maggiori di due. Anche effettuando le operazioni più significative in L.M. anziché in basic contribuise ad aumentare la velocità del procedimento.

Tutti questi accorgimenti sono stati adottati nel programma pubblicato e, a proposito del numero primo di Eulero, il nostro "C-64" impiega meno di mezzo minuto per dimostrarne la primalità.

Ovviamente non possono essere immessi numeri più lunghi di 10 cifre (fino a due elevato a 32) altrimenti il numero sarebbe arrotondato e le ultime cifre perse.

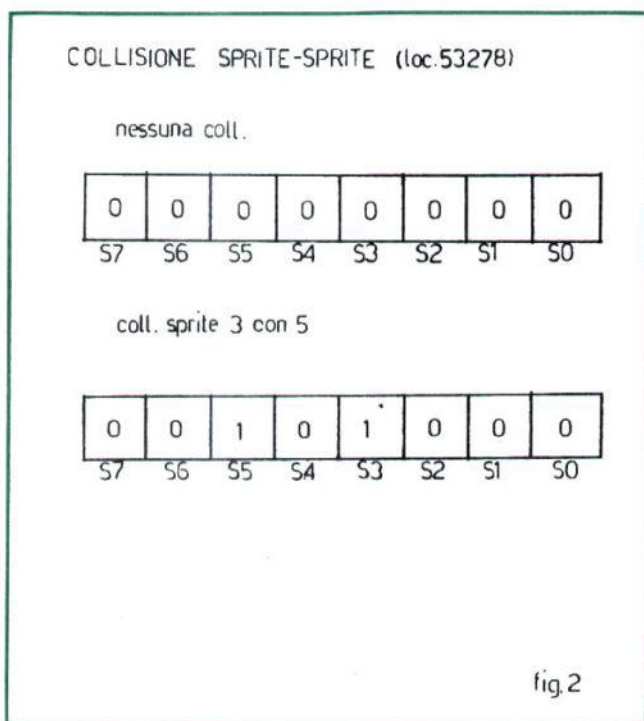
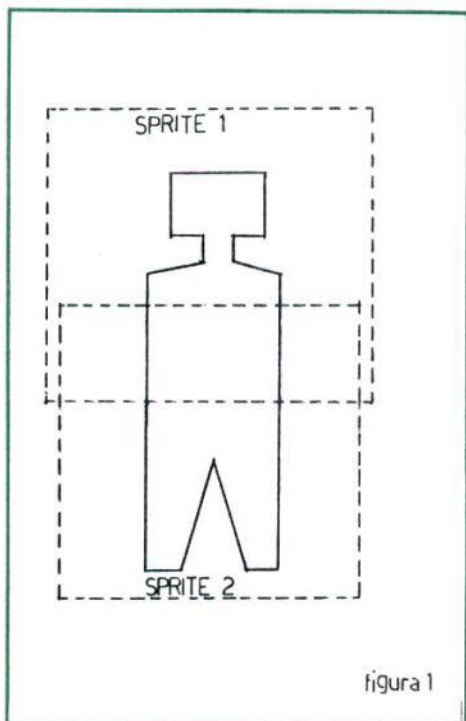
Flavio Molinari

```
100 REM ***      NUMERI PRIMI DI FERMAT
101 REM *** LISTATO PER QUALSIASI COMPUTER.
102 REM ***  DI FLAVIO MOLINARI (LAINATE)
103 :
120 B=2: REM BASE
130 INPUT"NUMERI PRIMI DA ...";L1:L1=INT(L1/2)*2+1
140 INPUT"                    A ...";L2
150 IFL2>65536THENPRINT"TROPPO GRANDI !!":GOTO130
155 PRINT
160 FORN=L1TOL2STEP2:Q=N:E=B:C=1
170 FORM=0TOLOG(N)*1.442:Q=Q/2:IFQ>INT(Q)THENC=C*E:C=C-INT(C/N)*N
180 E=E*E:E=E-INT(E/N)*N:Q=INT(Q):NEXT
190 IF C=B THEN PRINT N,
200 NEXT
READY.
```

```
100 REM ***            FATTORI PRIMI
101 REM *** LISTATO PER QUALSIASI COMPUTER
102 REM *** DI FLAVIO MOLINARI (LAINATE)
103 :
120 PRINT"■":II=49500
130 FORI=IITOII+103:READU:POKEI,U:NEXT
140 DATA 24,165,5,133,142,165,4,133,141,165,3,133,140,165,2,133,139
145 DATA 165,6,105,2,133,6,133,252,165,143,105,0,133,143,133,251
150 DATA 56,173,1,192,229,6,173,0,192,229,143,144,57
155 DATA 162,15,232,6,252,38,251,144,249,102,251,102,252,102,251,102,252
160 DATA 56,165,141,229,252,133,254,165,142,229,251,133,253,48,6
165 DATA 133,142,165,254,133,141,6,139,38,140,38,141,38,142,202,208,224
170 DATA 165,141,208,157,165,142,208,153,96
190 INPUT"■ BATTI IL NUMERO";M:M=ABS(M)
200 IFM>4.23E9THENPRINT" ■NUMERO TROPPO GRANDE !! ■":GOTO190
210 PRINT" ■"M"■ = ";
220 IFM/2>INT(M/2)THEN250
230 PRINT" 2 *";:M=M/2:IFM=2THEN370
240 GOTO220
245 REM:INIZIALIZZA ROUTINE IN L.M.
250 A=INT(M/256):B=INT(A/256):C=INT(B/256)
260 M1=M-A*256:M2=A-B*256:M3=B-C*256:M4=C
270 Q=INT(SQR(M)+1) :Q1=INT(Q/256):Q2=Q-Q1*256
280 POKE5,M4:POKE4,M3:POKE3,M2:POKE2,M1
290 POKE6,1:POKE143,0:POKE251,0:POKE252,0:POKE49152,Q1:POKE49153,Q2
300 SYS(II)
310 A=PEEK(143)*256+PEEK(6):IFM/A>INT(M/A)THEN370
320 REM:STAMPA FATTORE PRIMO
330 M=M/A:PRINTA;:IFM=1THENPRINT:GOTO190
340 PRINT"*";:IFA*A>MTHEN370
350 IFM/A=INT(M/A)THEN330
360 GOTO250
370 PRINTM:GOTO190
READY.
```

ABILITY GAME

figura 1

fig.2

Un gioco semplice e divertente utilissimo per comprendere la gestione delle collisioni tra gli sprite

Molti di voi, quelli che hanno provato a utilizzare sul CBM 64 gli sprite, si sono trovati in difficoltà nel realizzare un gioco in cui si dovrebbe colpire un uomo alla testa per realizzare, ad esempio, 100 punti, al busto per realizzarne 50 e alle gambe per totalizzarne 10.

Come molti sapranno nel 64 esistono due registri che controllano le collisioni sprite-sprite e le collissioni sprite-sfondo.

Questi registri hanno però un inconveniente: assumono infatti lo stesso valore sia che l'uomo venga colpito alla testa, al busto e alle gambe, e non sembra possibile differenziare le varie parti dello sprite per permettere di assegnare a queste diversi punteggi.

Un modo per ovviare all'inconveniente è quello di realizzare il nostro uomo non con un solo sprite, ma con due o più sprite in parte sovrapposti, in modo che il registro di collisione assuma valori diversi per ogni sezione del corpo e sia possibile differenziare i vari punteggi.

Ad esempio possiamo costruire con uno sprite la testa e il busto, e con un altro sprite ancora il busto e le gambe, come in figura 1. In questo modo il registro di collisione può assumere tre valori diversi, in quanto può venire colpito solo lo sprite 1, solo lo sprite 2 oppure tutti e due assieme, realizzando così il nostro scopo.

Naturalmente gli sprite sovrapposti possono anche essere più di due, permettendo così di differenziare molte parti di qualsiasi "cosa" che possa servire in un programma.

I registri degli sprite

Vediamo ora di analizzare l'uso dei due registri di collisione del 64. Gli sprite del 64 sono 8, numerati da 0 a 7. Come è possibile controllare la collisione di 8 sprite diversi utilizzando un solo registro? La risposta è relativamente semplice: un qualsiasi registro della memoria è formata da un byte, ossia da 8 bit (il bit, da BInary digiT, è la più piccola informazione immagazzinabile nella memoria di un computer.

Basta perciò associare ad ognuno di

fig.3

fig.4

fig.5

fig.6

questi bit uno sprite ed a portare il giusto bit al valore 1 ogni qualvolta vi sia una collisione.

Ad esempio se lo sprite 3 collide con lo sprite 5, i bit 3 e 5 del registro di collisione sprite-sprite (che è allocato alla locazione 53278) saranno posti a 1, permettendo così, tramite opportune istruzioni IF, di rilevare la collisione.

La stessa cosa succede nel caso in cui uno sprite collida con un carattere dello sfondo. In questo caso verrà messo a 1 il corrispondente bit del registro di collisione sprite-sfondo (loc. 53279).

Le figure 2, 3, 4, 5, 6 aiutano a capire il concetto esposto. In queste figure, SS è il registro di collisione sprite-sprite mentre SF è il registro di collisione sprite-fondo. Quando leggiamo i registri (con istruzioni del tipo PEEK) non vengono forniti i bit a 1 e quelli a 0, ma il corrispondente valore decimale. La conversione da binario a decimale viene mostrata in fig. 7. Il listato 1 è una dimostrazione di quanto detto finora.

Descrizione del gioco

Il gioco che proponiamo è una applicazione di questo sistema. Con la punta di un'astronave dovremo cercare di toccare il centro della base spaziale che si muove casualmente (o quasi) sullo schermo, evitando di toccare i bordi della stessa.

A mano a mano che il tempo passa, l'apertura di accesso alla base si fa sempre più stretta e la punta dell'astronave più corta, rendendo così l'agganciamento più difficile.

La base è costituita da due sprite: uno per il bordo e uno per il centro, mentre l'astronave è realizzata con un solo sprite.

Se la punta dell'astronave tocca il centro della base il registro di collisione sprite-sprite (53278) verrà portato al valore 6, mentre se tocca il bordo, il valore sarà 5. La figura 8 mostra queste due collisioni.

Il movimento dell'astronave è ottenuto mediante i tasti W, X, A, D. In queste pagine vengono pubblicate due versioni del gioco: una in BASIC, lentissima ma molto commentata, che serve soltanto per capire

come funziona il programma. L'altra è un misto di BASIC e di L.M. per permettere a coloro a cui il gioco piacesse di averne una versione sufficientemente veloce.

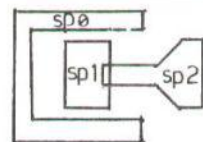
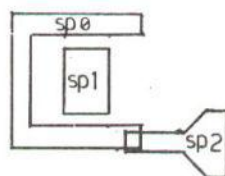
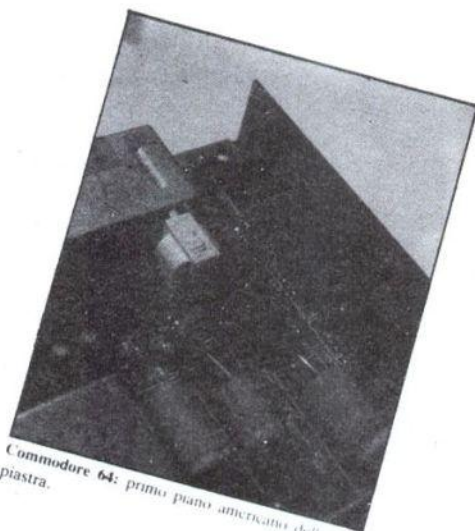
Nella versione BASIC+LM, il livello di difficoltà varia da "15" (per i principianti) a "0" per gli esperti.

Si raccomanda la solita attenzione nel digitare le routine in LM perché l'errore di un solo dato potrebbe portare al funzionamento irregolare del gioco o addirittura all'"impastamento" della macchina, costringendo a digitare nuovamente il listato.

Giancarlo Mariani

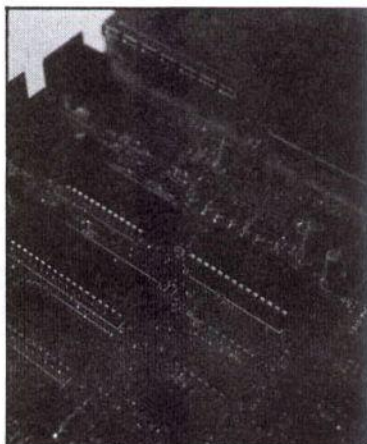
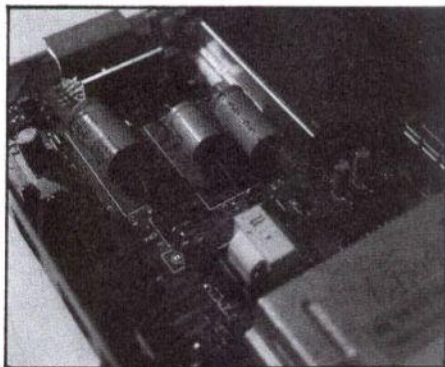
fig.7

fig. 8

```
1 REM ** ABILITY  GAME  BASIC **
2 REM ** BY MARIANI GIANCARLO **
3 REM **     (COMMODORE 64)    **
4 REM **     TASTI  A,D,W,X    **
5 :
15 POKE53269,0:PRINT"ATTENDERE PREGO..."
30 POKE2040,11:POKE2041,13:POKE2042,14
34 POKE53287,7:POKE53288,2:POKE53289,5:POKE53280,1:POKE53281,1
35 POKE53269,0:POKE53277,7:POKE53271,7
37 REM *** LEGGE DATI ***
40 FORK=0TO62:READA:POKE704+K,A:NEXT
50 FORK=0TO62:READA:POKE832+K,A:NEXT
52 FORK=0TO62:READA:POKE896+K,A:NEXT
55 PRINT"":GOSUB 7000
56 REM *** INIZIALIZZ. VARIABILI ***
57 CLR:TI$="000000":C$="000015":D$="000025"
58 A=150:B=51:GOSUB60000:U1=Q:M=53278
60 GOSUB60000:U2=Q:O1=53248:V1=O1+1:O2=O1+2:V2=O1+3:GOSUB900
61 N=PEEK(M):REM LEGGE REGISTRO DI COLLISIONE SPRITE PER AZZERARLO
62 IFTI$<=D$THENGOSUB2000
63 P=PEEK(197):B=B+(P=10)-(P=18):A=A+(P=9)-(P=23):REM TASTO PREMUTO?
64 IFA<50THENA=50:REM CONTROLLI DI MARGINE
65 IFA>200THENA=200
66 IFB<22THENB=22
67 IFB>255THENB=255
68 POKE53252,B:POKE53253,A:REM MUOVE ASTRONAVE
69 GOSUB500:N=PEEK(M):IFN<5THEN62:REM CONTROLLO ASSENZA COLLISIONI
70 POKE53269,0:TA$=TI$:IFN=6THEN300
72 REM TOCCATO IL BORDO (COLLIS.=5)
```

```
75 PRINT"HAI TOCCATO IL BORDO DELLA BASE."
80 PRINT"MI DISPIACE.....HAI PERSO"
85 PRINT"TEMPO:"TA$
90 PRINT"UN'ALTRA PARTITA? (S/N)"
100 GETA$:IFA$<>"S"ANDA$<>"N"THEN100
110 IFA$="N"THENEND
120 RUN
230 REM TOCCATO CENTRO (COLLIS.=6)
300 PRINT"BRAVISSIMO!!!!!"
310 PRINT"SEI RIUSCITO A TOCCARE IL SIMBOLO"
315 PRINT"COMMODORE....HAI VINTO!!!".
320 GOTO85
430 REM MUOVE BASE SPAZIALE
500 P1=PEEK(O1):P2=PEEK(V1)
502 P1=P1+U1:P2=P2+U2
510 IFP2<50THENP2=50:U2=-U2
515 IFP2>200THENP2=200:U2=-U2
520 IFP1<22THENP1=22:U1=-U1
525 IFP1>255THENP1=255:U1=-U1
550 POKEO1,P1:POKEV1,P2:POKEO2,P1:POKEV2,P2:RETURN
890 REM *** POSIZIONI INIZIALI ***
900 POKEO1,50:POKEV1,50:POKEO2,50:POKEV2,50
910 POKE53253,A:POKE53252,B:RETURN
1000 REM SPRITE 1
1005 DATA 0,0,0
1010 DATA 127,255,254,127,255,254
1015 DATA 127,255,254,112,0,0
1020 DATA 112,0,0,112,0,0,112,0,0
1025 DATA 112,0,0,112,0,0,112,0,0
1030 DATA 112,0,0,112,0,0,112,0,0
1035 DATA 112,0,0,112,0,0,112,0,0
1040 DATA 127,255,254,127,255,254
1045 DATA 127,255,254,0,0,0
1050 REM SPRITE 2
1055 DATA 0,0,0,0,0,0,0,0,0,0,0
1060 DATA 0,0,0,0,0,0,0,0
1065 DATA 1,255,224,1,243,224
1070 DATA 1,236,224,1,223,224
1075 DATA 1,236,224,1,243,224
1080 DATA 1,255,224
1085 DATA 0,0,0,0,0,0,0,0,0,0,0
1090 DATA 0,0,0,0,0,0,0,0
1100 REM SPRITE 3
1110 DATA 0,0,0,0,0,0,0,254
1120 DATA 0,1,252,0,1,252,0,15,248
1125 DATA 0,31,254,0,49,94,0,55,92
1130 DATA 127,241,24,127,245,216
1135 DATA 0,49,220,0,63,254,0,31,254
1140 DATA 0,15,248,0,1,252,0,1,252
1145 DATA 0,0,254,0,0,0,0,0,0,0,0
1900 REM *CONTROLLO PER RESTRINGERE L'ACCESSO ALLA BASE E PER
1910 REM *ACCORCIARE LA PUNTA DELL'ASTRONAVE
```

Commodore 64: primo piano americano della piastra.

```
2000 IFTI$=C$THEN2010
2005 GOTO2025
2010 POKE718,14:POKE721,14:POKE751,14:POKE754,14:POKE923,31:POKE926,31
2025 IFTI$=D$THENPOKE724,14:POKE727,14:POKE745,14:POKE748,14:POKE923,
     7:POKE926,7
2030 RETURN
6900 REM ** INIZIO **
7000 POKE53269,0:PRINT"ABILITY GAME BY MARIANI GIANCARLO"
7010 PRINT"TEL.0362/72565"
7020 PRINT"PREMERE UN TASTO PER INIZIARE"
7030 POKE198,0:WAIT198,1:PRINT"":POKE53269,7:RETURN
59900 REM NUM. CASUALE TRA -1 E 1 PER INIZIO MOVIM. BASE
60000 Q=(INT(RND(1)*3)-1):IFQ=0THEN60000
60005 RETURN
READY.
```

```
1 REM *** ABILITY GAME (VERSIONE L.M.)
2 REM *** BY MARIANI GIANCARLO
4 :
10 POKE53269,0:PRINT"ATTENDERE PREGO..."
51 FORT=0TO62:READA:POKE704+T,A:NEXT
52 FORT=0TO62:READA:POKE832+T,A:NEXT
53 FORT=0TO62:READA:POKE896+T,A:NEXT
54 IFPEEK(49152)=165ANDPEEK(49153)=197THEN56
55 FORT=0TO204:READA:POKE49152+T,A:NEXT
56 POKE53280,1:POKE53281,1:GOSUB7000:POKE2040,11:POKE2041,13:POKE2042,14
57 POKE53287,7:POKE53288,2:POKE53289,5
58 POKE53277,255:POKE53271,255
59 CLR:A=150:B=51:M=53278
60 GOSUB900:N=PEEK(M):TI$="000000"
63 SYS49345:N=PEEK(M):IFN>=253THEN70
64 GOTO63
70 POKE53269,0:TA$=TI$:IFN=254THEN300
75 PRINT"HAI TOCCATO IL BORDO DELLA BASE."
80 PRINT"MI DISPIACE.....HAI PERSO"
85 PRINT"TEMPO: "TA$
90 PRINT"UN'ALTRA PARTITA? (S/N)"
100 GETA$:IFA$<>"S"ANDA$<>"N"THEN100
110 IFA$="N"THENEND
120 RUN
300 PRINT"BRAVISSIMO!!!!!"
310 PRINT"SEI RIUSCITO A TOCCARE IL SIMBOLO"
315 PRINT"COMMODORE....HAI VINTO!!!"
320 GOTO85
900 POKE53248,50:POKE53249,50:POKE53250,50:POKE53251,50
910 POKE53252,B:POKE53253,A:RETURN
1000 REM SPRITE 1
1005 DATA 0,0,0
1010 DATA 127,255,254,127,255,254
1015 DATA 127,255,254,112,0,0
1020 DATA 112,0,0,112,0,0,112,0,0
```

```
1025 DATA 112,0,0,112,0,0,112,0,0
1030 DATA 112,0,0,112,0,0,112,0,0
1035 DATA 112,0,0,112,0,0,112,0,0
1040 DATA 127,255,254,127,255,254
1045 DATA 127,255,254,0,0,0
1050 REM SPRITE 2
1055 DATA 0,0,0,0,0,0,0,0,0,0,0,0
1060 DATA 0,0,0,0,0,0,0,0,0
1065 DATA 1,255,224,1,243,224
1070 DATA 1,236,224,1,223,224
1075 DATA 1,236,224,1,243,224
1080 DATA 1,255,224
1085 DATA 0,0,0,0,0,0,0,0,0,0,0,0
1090 DATA 0,0,0,0,0,0,0,0,0
1100 REM SPRITE 3
1110 DATA 0,0,0,0,0,0,0,0,254
1120 DATA 0,1,252,0,1,252,0,15,248
1125 DATA 0,31,254,0,49,94,0,55,92
1130 DATA 127,241,24,127,245,216
1135 DATA 0,49,220,0,63,254,0,31,254
1140 DATA 0,15,248,0,1,252,0,1,252
1145 DATA 0,0,254,0,0,0,0,0,0,0,0,0
4999 REM *** DATI LM ***
5000 DATA165,197,201,10,240,13,201,18
5010 DATA240,23,201,9,240,33,201,23
5020 DATA240,43,96,206,4,208,173,4
5030 DATA208,201,22,208,3,238,4,208
5040 DATA96,238,4,208,173,4,208,201
5050 DATA255,208,3,206,4,208,96,206
5060 DATA5,208,173,5,208,201,50,208
5070 DATA3,238,5,208,96,238,5,208
5080 DATA173,5,208,201,200,208,3,206
5090 DATA5,208,96
5100 DATA238,0,208,238,1,208,173,0
5110 DATA208,141,2,208,173,1,208,141
5120 DATA3,208,173,0,208,201,22,240
5130 DATA16,201,255,240,12,173,1,208
5140 DATA201,50,240,14,201,200,240,10
5150 DATA96,173,75,192,73,32,141,75
5160 DATA192,96,173,78,192,73,32,141
5170 DATA78,192,96
5500 DATA165,161,205,60,3,240,6,205
5510 DATA61,3,240,24,96,169,14,141
5520 DATA206,2,141,209,2,141,239,2
5530 DATA141,242,2,169,31,141,155,3
5540 DATA141,158,3,96,169,14,141,212
5550 DATA2,141,215,2,141,233,2,141
5560 DATA236,2,169,7,141,155,3,141
5570 DATA158,3,96
5600 DATA32,134,192,32,0,192,32,75
5610 DATA192,96
5640 DATA3,96
```

Commodore 64: particolare della piastra. Si può notare il generatore di suoni (SID) 6581.

Commodore 64: circuito di alimentazione e stabilizzazione

```
7000 PRINT"AB.GAME  BY MARIANI GIANCARLO"
7010 PRINT"TEL.0362/72565"
7015 INPUT"DIFFICOLTA' (1-250)";DI
7020 IFDI<1ORDI>250THEN7000
7025 POKE828,DI:POKE829,DI+3
7110 PRINT"":POKE53269,255
7120 POKE49227,238:POKE49230,238:RETURN
READY.
```

```
10 REM ABILITY GAME: INTRODUZIONE
20 REM LISTATO PER PROVARE I REGISTRI DI COLLISIONE
30 :
31 CS=53278:REM REGISTRO DI COLLISIONE TRA SPRITE E SPRITE
32 CF=53279:REM REGISTRO DI COLLISIONE TRA SPRITE E SFONDO
35 O0=53248:REM COORD. ORIZZONTALE SPRITE 0
36 V0=53249:REM COORD. VERTICALE SPRITE 0
37 O1=53250:REM COORD. ORIZZONTALE SPRITE 1
38 V1=53251:REM COORD. VERTICALE SPRITE 1
40 POKE 2040,11:REM SELEZ.BLOCCO DI MEM.11 (LOC.704).SPRITE 0
45 POKE 2041,11:REM IDEM COME SOPRA PER SPRITE 1
50 FOR K=704 TO 767:POKE K,255:NEXT K:REM RIEMPIE IL BLOCCO CON 255
60 POKE 53287,0:REM SELEZIONA IL COLORE NERO PER LO SPRITE 0
65 POKE 53288,7:REM SELEZIONA IL GIALLO PER LO SPRITE 1
70 O=100:V=100:REM COORDINATE DI PARTENZA
80 POKE53269,3:REM ACCENDE GLI SPRITE 0 E 1 (0000011: 1+2=3)
90 PRINT"";TAB(19);"A":REM SCRIVE UNA "A" SULLO SCHERMO
100 REM *** PRIMO CASO: NESSUNA COLLISIONE ***
110 POKE O0,30:POKE V0,110:REM POSIZIONA LO SPRITE 0 A 30,110
120 POKE O1,30:POKE V1,170:REM POSIZIONA LO SPRITE 1 A 30,170
130 GOSUB 10000
140 REM *** SECONDO CASO: COLLISIONE SPRITE CON SPRITE ***
150 POKE V1,120:REM SPOSTA SPRITE 1 SOPRA SPRITE 2
160 GOSUB 10000
170 REM *** TERZO CASO: COLLISIONE SPRITE 0 CON SFONDO ***
180 POKE O0,160:GOSUB 10000
190 REM *** QUARTO CASO: COLLISIONE SPRITE 1 CON SFONDO ***
200 POKE O0,30:POKEO1,160:GOSUB 10000
210 REM *** QUINTO CASO:COLLISIONE SPRITE 0 E 1 CON SFONDO ***
220 POKE O0,160:POKE V0,95:POKE V1,120:GOSUB 10000
230 REM *** FINE ***
235 POKE 53269,0:REM "SPEGNE" I 2 SPRITE
240 PRINT"FINE":END
9999 END
10000 REM SCRIVE SU VIDEO I DUE REGISTRI DI COLLISIONE
10002 A=PEEK(CS):A=PEEK(CF):REM AZZERA I DUE REGISTRI DI COLLIS.
10005 PRINT"COLLISIONE SPRITE CON SPRITE: ";PEEK(CS)
10010 PRINT"COLLISIONE SPRITE CON SFONDO: ";PEEK(CF)
10020 PRINT"PREMI UN TASTO"
10030 POKE198,0:WAIT198,1:RETURN
READY.
```

IL PLOTTER 1520

Tutte le istruzioni della periferica spiegate attraverso semplici esempi.

La 1520 è una stampante grafica a 4 colori. Ciò che interessa sottolineare è che, al momento dell'accensione, stampa 4 quadratini.

Se i pennini sono stati inseriti nel modo corretto, i colori dei quadratini devono essere, da sinistra a destra, blu — verde — rosso — nero. L'area tracciabile ha le dimensioni di 480 punti (numerati da 0 a 479) nel senso della larghezza (che chiameremo asse X), e +/-999 punti (numerati da 0 a +/- 998) nel senso della lunghezza (che chiameremo asse Y), con risoluzione di 0.2 mm.

Ciò significa che lo spostamento minimo del pennino è di 0.2 mm. La larghezza "utile" risulta perciò essere di 96 mm. (480*0.2) mentre la lunghezza "utile" di +/- 199.8 mm. (999*0.2).

Se volete farvi un'idea dell'area tracciabile, digitate il programmino di figura 1 assicurandovi, dapprima, di avere a disposizione almeno 25 cm. di carta.

Comandi Basic

I comandi Basic che è necessario usare di frequente col plotter sono i seguenti:

• OPEN: sintassi OPEN lfn, den, sa

Questo comando crea una relazione tra la periferica usata (il plotter, nel nostro caso) ed un certo file.

lfn: è il numero del file logico

dn: è il numero della periferica (nel nostro caso 6)

sa: è l'indirizzo secondario, cioè il modo in cui il plotter viene istruito per compiere un certo lavoro.

A pagina 20 del manuale d'uso, potete trovare una tabellina di corrispondenze tra i numeri dei file logici (lfn) e gli indirizzi secondari (sa).

• PRINT#: sintassi PRINT# lfn, <ar>

Ricordiamo esplicitamente che la parola PRINT, in questo caso, non può essere abbreviata col punto interrogativo (?).

Il senso di questo comando è analogo a quello di Print solo che l'output è riferito al plotter

lfn: è sempre il numero del file logico che deve essere prima "aperto" con l'istruzione Open

var: questo punto lo discuteremo in seguito

• CLOSE: sintassi CLOSE lfn

Chiude il file logico (lfn) precedentemente aperto.

Un semplice esempio dell'uso di questi 3 comandi lo potete avere digitando in maniera diretta (senza, cioè, numero di linea) le seguenti istruzioni che hanno il solo scopo di resettare il plotter:

```
OPEN 7, 6, 7     [ret]
PRINT#7          [ret]
CLOSE7           [ret]
```

• CMD: sintassi CMD lfn, <var>

Questo comando trasferisce la stampa dal video ad un file logico specificato (lfn).

Questa istruzione va usata, in genere, solo in modo diretto per ottenere, tra l'altro, il listato di un programma Basic.

Le istruzioni sono le seguenti:

```
OPEN4, 6         [ret]
PRINT#4          [ret]
CMD4             [ret]
LIST             [ret]
PRINT#4          [ret]
CLOSE4           [ret]
```

Volendo, si può stampare anche solo una parte di programma modificando l'istruzione LIST con l'aggiunta del numero di linea iniziale e finale relativi alla parte di programma che si vuole stampare, proprio come siete abituati listando su video.

Esempio: LIST 100-200

Il programma verrà listato usando 40 caratteri per linea.

Se avete già provato a listarne uno, magari di sole 100 linee, vi sarete accorti, guardandovi allo specchio, che nel frattempo siete invecchiati: è, infatti, necessario circa un quarto d'ora! Il fatto è che il plotter i caratteri se li deve costruire uno per uno. Fra l'altro, non riesce a costruirli nemmeno tutti!

I caratteri in reverse del modo virgolette e i caratteri grafici non coincidono con quelli del video. Un modo per dimezzare (circa) il tempo per un listato, è quello di stampare nel modo a 80 caratteri per linea consigliato solo a chi ha una buona vista oppure a chi ha uno spirito da martire.

Indirizzi secondari

Col comando OPEN abbiamo parlato di indirizzo secondario, affermando che questo è l'unico modo per istruire il plotter a compiere una certa operazione.

Più in dettaglio tali operazioni sono:

Operazioni	Indir. sec.
Stampa caratteri Ascii	0
Coordinate X, Y per grafici	1
Selezione colore	2
Selezione grandezza carattere	3
Selezione rotazione carattere	4
Selezione tratteggio	5
Selezione maiuscolo/minusc.	6
Reset del plotter	7

Per ogni operazione desiderata, occorrerà aprire (OPEN) un file e, di conseguenza, completare il comando PRINT# col numero del File aperto, al quale corrisponderà un certo indirizzo secondario (come vedremo più avanti).

Con il computer Commodore il numero massimo di File che si possono aprire contemporaneamente è 10. Dalla tabella precedente si vede che il numero di indirizzi secondari è 8, a cui corrispondono 8 File. Ciò significa che in uno stesso programma possiamo utilizzare tutte le operazioni che il plotter è in grado di svolgere.

Per evitare confusione, come suggerito sul manuale d'uso, è opportuno usare i seguenti numeri convenzionali per indicare i vari File:

Numero file	Ind. sec.	Esempio
4	0	OPEN 4, 6
1	1	OPEN 1, 6, 1
2	2	OPEN 2, 6, 2
3	3	OPEN 3, 6, 3
44	4	OPEN 44, 6, 4
5	5	OPEN 5, 6, 5
6	6	OPEN 6, 6, 6
7	7	OPEN 7, 6, 7

Riassumendo le operazioni necessarie per svolgere una certa operazione, sono le seguenti:

nn OPEN lfn, dn, sa
nm PRINT# lfn <,var>
mm CLOSE lfn

dove nn, nm, mm sono numeri di linea.

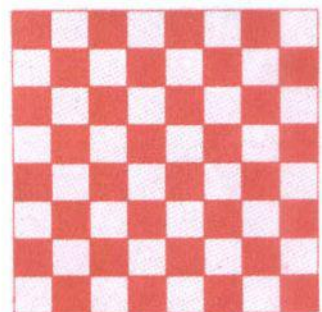
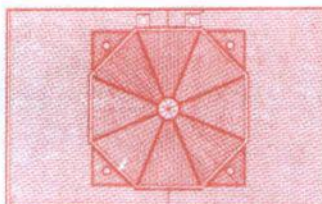
Esaminiamo ora in dettaglio ciascun indirizzo secondario (SA sta per "secondary address").

- SA=0: stampa i caratteri Ascii

Il formato del comando PRINT# è il seguente: PRINT#4, "dati"

Il numero 4 è quello del file logico associato a questo indirizzo secondario (vedi tabella precedente). Tutto quello che si trova tra virgolette, verrà stampato così com'è.

Per chiarirvi le idee, potete digitare il programma di pagina 21 del manuale d'uso. Ora provate ad aggiungere questa linea: 145 PRINT#4.

Le scritte precedenti verranno separate da una riga vuota, proprio come succede

quando usate l'istruzione Print.
In qualunque posizione si trovi il pennino, dopo Print#4 si riporta all'inizio e scende di quante righe volete.

Un'ultima cosa da sottolineare, è che l'indirizzo secondario 0 è il valore di DEFAULT, cioè può anche essere omesso nell'istruzione Open, poiché verrà assunto automaticamente.
Al contrario, se dimenticate di specificare l'indirizzo secodario (SA) in Open, questo verrà assunto come 0.

- SA=6: seleziona maiuscolo/minusc.

Il formato è il seguente:
PRINT#6, n

In cui "n" può assumere i valori "0" oppure "1".

Il valore 0 corrisponde alla stampa normale, ossia caratteri maiuscoli. All'accensione, il plotter si trova in tali condizioni. Il valore 1 corrisponde, invece, ai caratteri minuscoli.
Praticamente, l'effetto dell'istruzione PRINT#6, 1 è analogo a quello che ottenete per il video premendo il tasto [C] +SHIFT.

Se omettete "n", verrà assunto automaticamente il valore 0 (valore di Default). L'effetto di questo SA lo potete vedere digitando il programma di figura 2. Se volete evidenziare i caratteri maiuscoli e minuscoli in una stessa parola, potete operare portando il video a caratteri minuscoli e digitando il programma di figura 3. Il modo ivi descritto non è l'unico per ottenere caratteri maiuscoli e minuscoli.

Dopo aver fatto girare un programma con un'istruzione PRINT#6, 1 il plotter rimane nel modo minuscolo. Se lo volete riportare nelle condizioni normali, digitate, in modo diretto, queste istruzioni:

OPEN 6, 6, 6
PRINT# 6 (0) (non occorre digitare ,0)
CLOSE 6

- SA=2: seleziona dei colori

Il formato è il seguente:
PRINT#2, C

In cui "C" è il numero del colore desiderato. Più precisamente:

NERO	C=0
BLU	C=1
VERDE	C=2
ROSSO	C=3

Questa corrispondenza è vera solo se i pennini sono stati inseriti nella giusta posizione. Se non usate affatto questo Ind. Sec., il plotter stampa automaticamente in nero.

Utilizzando nuovamente il programma di pagina 21 (manuale d'uso), apportate queste modifiche:

```
125 OPEN 2, 6, 2
135 PRINT#2, C : REM*cambia colore*
145 PRINT#4    : REM*salta una riga*
146 C=C+1      : REM*incrementa C*
170 CLOSE 2
```

- SA=3: seleziona grandezza carattere

Il formato è il seguente:
PRINT#3, M

Con questo indirizzo, si può scegliere tra 4 dimensioni diverse per stampare i caratteri Ascii.

M può assumere valori tra 0 e 3, secon-

do la seguente tabella:

Numero	Dimensione carattere
0	80 caratteri per linea
1	40 caratteri per linea
2	20 caratteri per linea
3	10 caratteri per linea

L'esempio di pagina 30, penso sia più che sufficiente come dimostrazione.

Tenete comunque presente che, se volete 'scrivere' qualche parola o frase col plotter, bisogna sempre richiamare SA=0, e di conseguenza il File logico numero 4: questo è fatto nella linea 120 (dove lo 0 è omesso: vi ricordate perché?... Va bene, rispondo io: l'indirizzo secondario numero 0 è l'indirizzo di DEFAULT).

Fatto questo, tutto quello che volete scrivere deve essere preceduto da PRINT#4, (...) (cfr. linea 170).
Fissate ora l'attenzione sulla linea 195.

Questa riporta il plotter nelle condizioni normali, cioè 40 caratteri per linea (M=1 valore di Default per questo indirizzo). Se questa linea fosse omessa, tutte le volte che vorreste scrivere qualcosa, questa avrebbe le dimensioni dell'ultima M impostata (nel nostro caso M=3: 10 caratteri per linea).

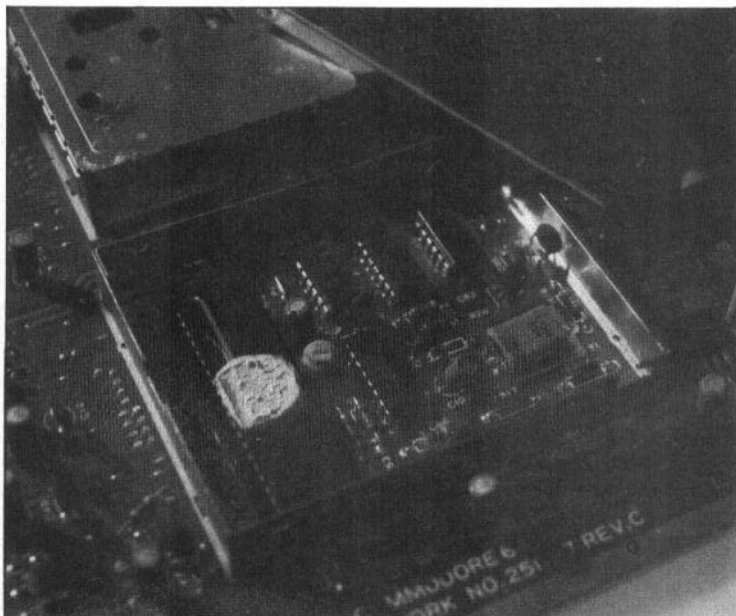
In precedenza avevo detto che un modo per dimezzare il tempo di un listato era quello di usare 80 caratteri per linea.
Se volete provare, dopo aver digitato un programmino di prova, usate queste istruzioni in modo diretto:

```
OPEN4, 6
OPEN3, 6, 3
PRINT#4
PRINT#3, 0
CMD4
LIST
PRINT#4
PRINT#3, 1: REM * riporta nelle condizioni normali*
CLOSE4
CLOSE3
```

- SA=7: resetta il plotter

  Formato:

  PRINT#7

Questo indirizzo, come già visto in precedenza, serve per resettare il plotter. Nonostante le apparenze, devo dire che questo comando è veramente importante.

Commodore 64: il generatore video.

Quando si fa girare un programma che usa, per esempio, vari colori e varie dimensioni per i caratteri, le condizioni finali rimangono quelle impostate per ultime dal programma.
Per un ritorno alle condizioni normali, bisogna operare con le istruzioni indicate nei programmi precedenti.

Per evitare tutti questi fastidi, basterà scrivere, alla fine del programma, l'istruzione PRINT#7 preceduta da OPEN 7, 6, 7 e seguita da CLOSE 7.

Automaticamente, infatti, il plotter verrà resettato, cioè tutti gli Indirizzi Secondari assumeranno il valore di DEFAULT; a conferma di questo, verranno anche stampati i soliti 4 quadratini.

- SA=1: plot X, Y

  Formato:

  PRINT#1, "Sotto-Comando" (,X, Y)

Questo indirizzo rappresenta il "cuore" del plotter: fornisce, infatti, la possibilità di controllare tutti i movimenti del pennino all'interno dell'area tracciabile discussa all'inizio.

Molti esempi d'uso di questa istruzione li trovate sul manuale e nei programmi pubblicati sul n. 16 di questa rivista. Per chi li trovasse troppo complicati, cercheremo ora di illustrare i vari Sotto-Comandi in modo semplice.

I Sottocomandi a pennino sollevato

- Sotto-Comando H: PRINT#1, "H"

Riporta il pennino al punto di partenza, (pennino SOLLEVATO) dato dalle coordinate X=0, Y=0.

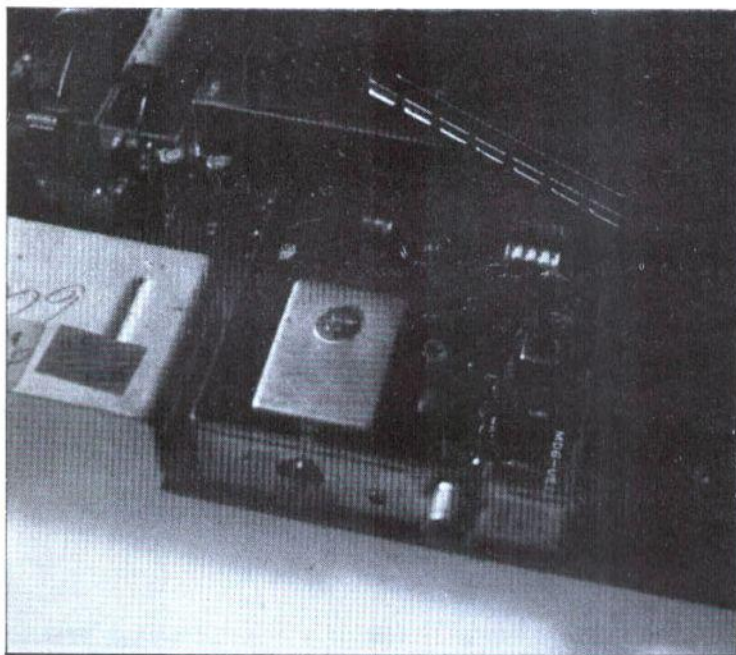
Inizialmente, il punto (0,0) è nella posizione di HOME (casa=ORIGINE ASSOLUTA), cioè nella posizione che assume il pennino al momento dell'accensione.

Vedremo che esiste la possibilità di spostare il punto (0,0) in qualsiasi posizione dell'area tracciabile.

- Sotto-Comando M: PRINT#1, "M", X, Y

Sposta il pennino nella posizione X, Y rispetto all'Origine Assoluta (home) Pennino SOLLEVATO.

Provate a digitare il seguente programma:

```
10 OPEN1, 6, 1
20 PRINT#1, "M", 240, 0
```

Commodore 64: la parte di radio frequenza.

300 PRINT#1, "H"
500 CLOSE1

Visto cosa succede? Praticamente niente! Il pennino si sposta al centro dell'asse X (in effetti, il centro è 239.5) e poi ritorna in Home.

Aggiungete la linea:
30 PRINT#1, "I"

Il pennino ora si ferma nella posizione centrale. La linea 30 (PRINT#1, "h") comunica al pennino di tornare nell'origine (0,0), solo che ora il punto (0,0) si trova nella posizione X=240, Y=0 rispetto all'Origine Assoluta.

- Sotto-Comando I: PRINT#1, "I"

Fissa la nuova origine (ORIGINE RELATIVA Xo, Yo) nel punto X, Y determinato dal sotto-comando "M": nel nostro caso X=240, Y=0. Come detto nell'introduzione, l'asse X ha a disposizione 480 punti (da 0 a 479): essendo ora l'origine in X=240, potete muovere il pennino verso sinistra al massimo di 240 punti e verso destra di 239.
Se l'origine relativa fosse in X=100, Y=0 si avrebbero a disposizione 100 punti a sinistra e 379 a destra, e così via.

- Sotto-Comando R: PRINT#"R", X, Y

Porta il pennino nella posizione X, Y relativa all'Origine Relativa Xo, Yo (pennino SOLLEVATO).

Sottocomandi a pennino abbassato

I Sotto-Comandi visti finora, sono tutti sotto-comandi a "pennino sollevato", (pen up — move), cioè vengono eseguiti senza che sulla carta sia tracciato niente.
Esaminiamo ora i comandi a "pennino abbassato" (pen down — draw).

- Sotto-Comando J: PRINT#1, "J", X, Y

Traccia la posizione X, Y rispetto all'origine relativa Xo, Yo.
Provate a sostituire la linea 30 con:
30 PRINT#1, "J", 100, 0

- Sotto-Comando D: PRINT#1, "D", X, Y

Traccia la posizione X, Y rispetto all'origine assoluta, partendo dalla posizione in cui si trova attualmente il pennino. Sostituite la linea 30 con:
30 PRINT#1, "D", 100, 0
Ora aggiungete le linee:
11 OPEN4, 6
400 PRINT#4
510 CLOSE4

La linea 400, se vi ricordate, serve per saltare una riga.
Tutte le volte che il programma incontra questa istruzione, il pennino viene riportato nella posizione di HOME: può quindi servire per resettare il plotter. Attenzione però che questo tipo di reset vale solo per SA=1. Il reset totale si fa solo con l'istruzione PRINT#7, che comporta, come già detto, la stampa dei 4 quadratini.

Applicazioni pratiche

Proviamo ora a costruire un cerchio applicando i vari sotto-comandi. Premetto subito che per costruire un cerchio, occorrono delle conoscenze di trigonometria che probabilmente non tutti hanno dato che questa parte della matematica viene affrontata solo negli anni terminali delle scuole superiori.

Tanto per cominciare, potete digitare il programma di fig. 5 e farlo girare.

La parte "matematica" è data dalle linee 100, 110, 120, 130.
In linea 100, i numeri 0, 360 e 10 sono GRADI. La linea 110 trasforma i gradi in RADIANTI, usati poi nelle linee 120, 130 come argomento delle funzioni SENO (SIN) e COSENO (COS).
Ricordiamo che il CBM 64 riconosce solo i radianti come unità di misura per gli angoli.

Le linee 10, 20, 30 si occupano dell'apertura dei file associati ai SA numerati con 0, 1, 7.

Le linee 200, 210, 220 si occupano della chiusura dei file precedentemente aperti.

La linea 40 porta il pennino nel punto X=240, Y=240 rispetto alla posizione di HOME.

La linea 50 fissa la nuova origine (origine relativa Xo, Yo) in questo punto.

L'origine relativa rappresenta il centro del cerchio. È, in altre parole, il punto in cui mettete la punta del compasso quando volete tracciare una circonferenza. La linea 150 è la linea principale: traccia i punti X, Y (dati in 120 e 130) rispetto al centro. Per capire la differenza tra "J" e "D", sostituite "D" al posto di "J" in li-

nea 150: D traccia i punti X, Y rispetto a HOME.

La linea 140 porta il pennino in posizione X, Y (rispetto al centro) quando l'angolo è di zero (quindi pennino sollevato). Provate a toglierla e vedete cosa succede! La linea 170 riporta il pennino nell'origine (centro del cerchio). Per finire, la linea 180 fa scendere il pennino di 7 righe rispetto al centro e la linea 190 resetta il plotter.

Il semplice programma di figura 5 può fare molte altre cose.
Cambiando lo STEP di linea 100 potete ottenere tutti i poligoni che volete. Provate, ad esempio, con i valori 30, 45, 90. In linea 120 e 130, quei *100 che compaiono rappresentano i raggi. Per ingrandire il cerchio, potete aumentare questi valori fino ad un massimo di 239.
Se volete tracciare ELLISSI, i valori dei due raggi devono essere diversi tra loro come, ad esempio, 100 e 50.

Per complicare un po' le cose, aggiungete la linea:
90 FOR K=100 TO 10 STEP 10
165 NEXT

e variate le linee 120, 130 come segue:
120 X=COS (II) *K
130 Y=SIN (II) *K

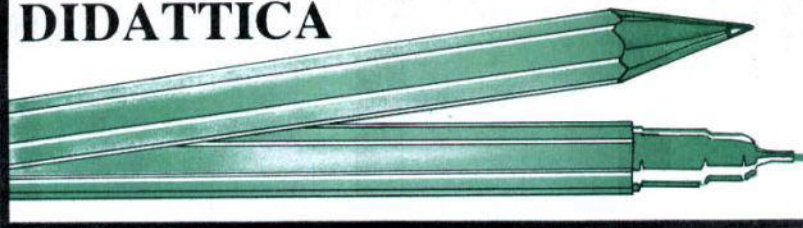
Giancarlo Castagna

```
10 REM *** TRACCIA AREA UTILE ***
15 REM ***  CON PLOTTER 1520   ***
20 OPEN1,6,1
30 PRINT#1,"D",0,998
40 PRINT#1,"D",479,998
50 PRINT#1,"D",479,-998
60 PRINT#1,"D",0,-998
70 PRINT#1,"D",0,0
80 CLOSE1
READY.
```

```
100 REM *** MAIUSCOLE & MINUSCOLE ***
101 :
120 OPEN4,6 : REM * INIZIAL. ASCII *
125 OPEN6,6,6 : REM * INIZIAL. MAIUS/MINUS *
130 FOR I = 1 TO 2
140 PRINT#4,"1520 PRINTER PLOTTER"
150 NEXT
160 PRINT#4 : REM * SALTA UNA RIGA *
170 PRINT#6,1 : REM * IMPOSTA MINUS *
180 FOR I = 1 TO 2
190 PRINT#4,"1520 PRINTER PLOTTER"
200 NEXT
210 CLOSE 4 : CLOSE 6 : REM * CHIUSURA FILE *
READY.
```

```
100 REM *** MAIUS + MINUS ***
101 :
120 OPEN4,6 : REM * INIZIAL. ASCII *
125 OPEN6,6,6 : REM * INIZIAL. MAIUS/MINUS *
170 PRINT#6,1 : REM * IMPOSTA MINUS *
180 FOR I = 1 TO 2
190 PRINT#4,"1520 ┐RINTER ┐LOTTER" : REM * SHIFT + P *
200 NEXT
210 CLOSE 4 : CLOSE 6
READY.
```

```
 1 REM * DEMO PLOTTER COMMODORE 1520 *
 2 REM *   ESEMPIO SA = 0,2,3,6,7      *
 3 :
 10 OPEN4,6    : REM * INIZIAL. ASCII *
 20 OPEN3,6,3 : REM * INIZIAL. DIMENSIONE CARATTERE *
 30 OPEN2,6,2 : REM * INIZIAL. COLORE *
 40 OPEN6,6,6 : REM * INIZIAL. MAIUS/MINUS *
 50 OPEN7,6,7 : REM * INIZIAL. RESET *
 100 PRINT#4:PRINT#4,"CONDIZIONI NORMALI"
 110 PRINT#2,1 : REM * CAMBIA COLORE *
 120 PRINT#4:PRINT#4,"ORA SONO BLU"
 130 PRINT#2,2 : REM * CAMBIA COLORE *
 140 PRINT#3,0 : REM * CAMBIA DIMENS. *
 145 PRINT#4
 150 PRINT#4,"ORA SCNO VERDE E STAMPO 80 CAR. PER LINEA"
 160 PRINT#2,3 : REM * CAMBIA COLORE *
 170 PRINT#3,2 : REM * CAMBIA DIMENS. *
 180 PRINT#4:PRINT#4,"SONO ROSSO 20"
 190 PRINT#3,3 : REM * CAMBIA DIMENS. *
 200 PRINT#4:PRINT#4,"MASSIMO 10"
 210 L=L+1
 220 IF L<2 THEN PRINT#6,1 :PRINT#3,1 :PRINT#2,0 :GOTO100
 230 REM * LA LINEA 220 RIPRISTINA IL NERO
 235 REM * E 40 CARATTERI PER RIGA
 240 REM * E PASSA AL MINUSCOLO *
 300 PRINT#7 : REM * RESETTA IL PLOTTER *
 500 CLOSE 4 : CLOSE 3 : CLOSE 2 : CLOSE 6: CLOSE 7 :REM *CHIUSURA FILE *
READY.
```

```
1 REM * COSTRUZIONE CERCHIO *
2 :
10 OPEN4,6 : REM * INIZIAL. ASCII *
20 OPEN1,6,1 : REM * INIZIAL. PLOT *
30 OPEN7,6,7 : REM * INIZIAL. RESET *
40 PRINT#1,"M",240,-240 : REM * PENNINO SOLLEVATO *
50 PRINT#1,"I" : REM * FISSA NUOVA ORIGINE *
100 FOR I=0 TO 360 STEP 10
110 II=(I/180)*π : REM * RADIANTI *
120 X=COS(II)*100
130 Y=SIN(II)*100
135 REM * PENNINO SOLLEVATO SE L'ANGOLO = 0
140 IF I=0 THEN PRINT#1,"R",X,Y
150 PRINT#1,"J",X,Y : REM * TRACCIA CERCHIO *
160 NEXT
170 PRINT#1,"H" : REM * PENNA NELL'ORIGINE (SOLLEVATA)
180 FOR I=1 TO 7 : PRINT#4 :NEXT : REM * SALTA 7 RIGHE
190 PRINT#7 : REM * RESETTA IL PLOTTER *
200 CLOSE 4 : REM * CHIUSURA FILE 4 *
210 CLOSE 1 : REM * CHIUSURA FILE 1 *
220 CLOSE 7 : REM * CHIUSURA FILE 7 *
READY.
```

LINGUAGGIO MACCHINA

Nelle puntate precedenti abbiamo cercato di fare un po' di luce sulla gestione interna del 6502, il microprocessore utilizzato dal Commodore 64 (nella versione 6510) e dal Vic 20. Ma quello che conosciamo, non ci permette ancora di creare programmi di una certa utilità ed interesse (o per lo meno richiede particolari sforzi ed una notevole quantità di memoria impiegata): si tratta di approfondire il discorso relativo agli indirizzamenti. Senza nessuna pretesa di riproporre ciò che è già stato detto, penso valga la pena di citare i 4 indirizzamenti che conosciamo:

- Implicito (AA= TAX)
- Immediato (A9 00 = LDA #$00)
- Assoluto (8D 00 04 = STA $0400)
- Relativo (D0 03 = BNE $+03)

Questi ci consentono di leggere il contenuto di una locazione di memoria, di modificarla e di eseguire salti condizionati. Proviamo ora a creare un semplice programma che trasferisca sullo schermo un certo messaggio i cui valori ASCII sono contenuti in memoria.

Il programma in questioe è quello di figura 1 (relativo al C 64, ma identico a quello di figura 1 bis, per il VIC 20). La parte iniziale (da 033A a 0356) costituisce la necessaria inizializzazione del programma. Il perché verrà spiegato in seguito. Successivamente l'accumulatore viene caricato con il contenuto della locazione di memoria a partire dalla quale sono inseriti i dati. Quindi il dato (in codice ASCII) viene posto nella casella in alto a sinistra nello schermo. Le successive due istruzioni (A9 00 e 8D 00 D8) settano il colore del carattere appena visualizzato. Per rendere ciclico il programma, si deve adesso modificare il low byte dei tre registri che fanno diretto riferimento alla memoria ed allo schermo (EE 58 03, ecc.)

Il loop si chiude con un paragone , ne-

```
033A A9 04       LDA#$04
033C 8D 5C 03    STA $035C
033F A9 03       LDA#$03
0341 8D 59 03    STA $0359
0344 A9 71       LDA#$71
0346 8D 58 03    STA $0358
0349 A9 D8       LDA#$D8
034B 8D 61 03    STA $0361
034E A9 00       LDA#$00
0350 8D 5B 03    STA $035B
0353 8D 60 03    STA $0360
0356 AA          TAX
0357 AD 71 03    LDA $0371
035A 8D 00 04    STA $0400
035D A9 00       LDA#$00
035F 8D 00 D8    STA $D800
0362 EE 58 03    INC $0358
0365 EE 5B 03    INC $035B
0368 EE 60 03    INC $0360
036B E8          INX
036C E0 18       CPX#$18
036E D0 E7       BNE $0357
0370 60          RTS
0371 43 4F 4D 4D 4F 44
0377 4F 52 45 20 43 4F
037D 4D 50 55 54 45 52
0383 20 43 4C 55 42 20
```

Figura 1

```
033A A2 00       LDX#$00
033C BD 4D 03    STA $034D,X
033F 9D 00 04    STA $0400,X
0342 A9 00       LDA#$00
0344 9D 00 D8    STA $D800,X
0347 E8          INX
0348 E0 18       CPX#$18
034A D0 F0       BNE $033C
034C 60          RTS
034D 43 4F 4D 4D 4F 44
0353 4F 52 45 20 43 4F
0359 4D 50 55 54 45 52
035F 20 43 4C 55 42 20
```

Figura 2

cessario per valutare il numero esatto di caratteri da trasferire, e con un salto in caso di mancato raggiungimento di tale numero (E0 18 e D0 E7). Si capisce quindi il perché si renda necessaria l'inizializzazione, senza la quale il programma, che in pratica si automodifica, non avrebbe potuto girare più di una volta consecutivamente senza ingenerare errori e cattivo funzionamento.

Il programma non è certo semplice, ma gli "architetti" del 6502 hanno previsto tale evenienza ed hanno dotato il microprocessore di un particolare indirizzamento: l'indirizzamento indicizzato. Vi ricordate il programma collocato alla fine della puntata precedente (e qui riproposto in figura 2 e 2 bis)? Se l'avete provato, vi sarete già resi conto che esegue lo stesso lavoro del programma appena descritto in modo più semplice e decisamente più breve.

Il "cuore" è costituito dalle due nuove istruzioni BD xx xx e 9D xx xx (LDA $0000,X e STA #0000,X); la loro particolarità è quella di sommare al valore assoluto che segue l'istruzione stessa il valore del registro X (registro ad 8 bit analogo ad A).

Per esempio, supponiamo di avere il seguente programma:

```
AZ 05       LDX #$05
BD 00 04    LDA  $0400,X
60          RTS
```

L'accumulatore verrà caricato con il contenuto della locazione $(0400 + 05) = $0405. Dunque, il registro X viene sommato al valore assoluto, e solo allora viene letta la locazione risultante. Analogo ragionamento vale per 9D xx xx, che, invece di leggere, modifica il contenuto della locazione risultante.

A tutto questo discorso aggiungiamo l'incremento e la comparazione del registro X con il valore assoluto ed otteniamo il programma di figura 2. Due aspetti importanti da sottolineare sono i seguenti:

- IL valore assoluto dell'indirizzamento rimane sempre lo stesso; quindi il programma non si automodifica e non ha bisogno di inizializzazione.
- Il 6402 tiene conto anche del riporto, vale a dire che, se nel corso del programma si supera una pagina di memoria, l'indicizzazione provvede a tutto. Esempio: $(7BFE + 05) dà automaticamete $7C04.

Per quanto riguarda il trasferimento di blocchi con più di 256 locazioni di memoria, si può dare un'occhiata al programma di figura 3. Esso provvede a trasferire sullo schermo le istruzioni Basic ed i relativi messaggi di errore. La sua struttura non è molto differente da quella del primo programma presentato, pur sfruttando l'indicizzazione. La differenza è che invece di alterare il low byte della locazione di memoria interessata, viene fatto variare il high byte.

```
033A A9 04     LDA#$04
033C 8D 7B 03  STA $037B
033F A9 A0     LDA#$A0
0341 8D 78 03  STA $0378
0344 A9 D8     LDA#$D8
0346 8D 80 03  STA $0380
0349 A9 00     LDA#$00
034B 8D 77 03  STA $0377
034E 8D 7A 03  STA $037A
0351 8D 7F 03  STA $037F
0354 A0 03     LDY#$03
0356 A2 00     LDX#$00
0358 20 76 03  JSR $0376
035B E8        INX
035C D0 FA     BNE $0358
035E EE 7B 03  INC $037B
0361 EE 78 03  INC $0378
0364 EE 80 03  INC $0380
0367 8B        DEY
0368 D0 EC     BNE $0356
036A 20 76 03  JSR $0376
036D A2 E8     LDX#$E8
036F 20 76 03  JSR $0376
0372 CA        DEX
0373 D0 FA     BNE $036F
0375 60        RTS
0376 BD 00 A0  LDA $A000,X
0379 9D 00 04  STA $0400,X
037C A9 00     LDA#$00
037E 9D 00 D8  STA $D800,X
0381 60        RTS
```

Figura 3

Per riempire lo schermo del Commodore 64 sono necessari 3 cicli da 256 byte l'uno, più un semiciclo (chiamato di offset) da 232 byte (256 * 3 + 232 = 1000) per il conto dei cicli viene usato il registro Y (analogo ad A e X), che viene decrementato ogni 256 byte trasferiti, mentre l'inizializzazione si rende necessaria per impostare i corretti valori negli high bytes modificati dal programma stesso. L'istruzione 20 xx xx (JSR xx xx) consente di fare un salto ad un determinato sottoprogramma e di ritornare al punto di partenza quando incontra l'istruzione 60 (RTS). Ma anche questi limiti possono essere facilmente superati e vedremo il metodo usato in un prossimo articolo.

Simone Bettol

```
10 REM  LINGUAGGIO MACCHINA  (SIMONE BETTOLA)
20 :
100 PRINT"[clr]FA APPARIRE IN ALTO A SINISTRA"
110 PRINT"SULLO SCHERMO DEL COMMODORE 64"
120 PRINT"LA SCRITTA 'COMMODORE COMPUTER"
130 PRINT"CLUB' SENZA USARE L'INDICIZZAZIONE"
140 PRINTCHR$(14)
200 AD=826:AD$="[crsr]SYS 826"
210 READA$:IFVAL(A$)<0THENPRINTAD$:END
220 X1=ASC(LEFT$(A$,1)):X2=ASC(RIGHT$(A$,1))
230 IFX1>57THENX1=X1-55:GOTO250
240 X1=X1-48
250 IFX2>57THENX2=X2-55:GOTO270
260 X2=X2-48
270 POKEAD,X1*16+X2:AD=AD+1:GOTO210
300 DATA A9,04,8D,5C,03,A9,03,8D
310 DATA 59,03,A9,71,8D,58,03,A9
320 DATA D8,8D,61,03,A9,00,8D,5B
330 DATA 03,8D,60,03,AA,AD,71,03
340 DATA 8D,00,04,A9,00,8D,00,D8
350 DATA EE,58,03,EE,5B,03,EE,60
360 DATA 03,E8,E0,18,D0,E7,60,43
370 DATA 4F,4D,4D,4F,44,4F,52,45
380 DATA 20,43,4F,4D,50,55,54,45
390 DATA 52,20,43,4C,55,42,20,00
400 DATA -1
READY.
```

```
10 REM LINGUAGGIO MACCHINA
20 :
100 PRINT"FA APPARIRE IN ALTO A SINISTRA"
110 PRINT"SULLO SCHERMO DEL COMMODORE 64"
120 PRINT"LA SCRITTA 'COMMODORE COMPUTER"
130 PRINT"CLUB' SFRUTTANDO L'INDICIZZAZIONE"
140 PRINTCHR$(14)
200 AD=826:AD$="SYS 826"
210 READA$:IFVAL(A$)<0THENPRINTAD$:END
220 X1=ASC(LEFT$(A$,1)):X2=ASC(RIGHT$(A$,1))
230 IFX1>57THENX1=X1-55:GOTO250
240 X1=X1-48
250 IFX2>57THENX2=X2-55:GOTO270
260 X2=X2-48
270 POKEAD,X1*16+X2:AD=AD+1:GOTO210
300 DATA A2,00,BD,4D,03,9D,00,04
310 DATA A9,00,9D,00,D8,E8,E0,18
320 DATA D0,F0,60,43,4F,4D,4D,4F
330 DATA 44,4F,52,45,20,43,4F,4D
340 DATA 50,55,54,45,52,20,43,4C
350 DATA 55,42,20,00,          -1
READY.
```

```
10 REM LINGUAGGIO MACCHINA
20 :
100 PRINT"STAMPA SULLO SCHERMO DEL "
110 PRINT"COMMODORE 64 LE ISTRUZIONI "
120 PRINT"BASIC ED I MESSAGGI DI ERRORE "
130 PRINT"SFRUTTANDO L'INDICIZZAZIONE"
140 PRINTCHR$(14)
200 AD=826:AD$="SYS 826"
210 READA$:IFVAL(A$)<0THENPRINTAD$:END
220 X1=ASC(LEFT$(A$,1)):X2=ASC(RIGHT$(A$,1))
230 IFX1>57THENX1=X1-55:GOTO250
240 X1=X1-48
250 IFX2>57THENX2=X2-55:GOTO270
260 X2=X2-48
270 POKEAD,X1*16+X2:AD=AD+1:GOTO210
300 DATA A9,04,8D,7B,03,A9,A0,8D
310 DATA 78,03,A9,D8,8D,80,03,A9
320 DATA 00,8D,77,03,8D,7A,03,8D
330 DATA 7F,03,A0,03,A2,00,20,76
340 DATA 03,E8,D0,FA,EE,7B,03,EE
350 DATA 78,03,EE,80,03,88,D0,EC
360 DATA 20,76,03,A2,E8,20,76,03
370 DATA CA,D0,FA,60,BD,00,A0,9D
380 DATA 00,04,A9,00,9D,00,D8,60
390 DATA -1
READY.
```

SALVATAGGIO/CARICAMENTO DELLA PAGINA GRAFICA

Ecco finalmente esaudita una delle richieste che mi sono sentito rivolgere da parte di vari lettori.

Non potete salvare su nastro (o su disco) i propri "capolavori", disegnati con le mie routine, era infatti un grosso handicap per tutti coloro che non conoscono approfonditamente il Sistema Operativo e la configurazione di memoria del nostro CBM 64. Effettivamente, dato che la pagina grafica è situata in una posizione particolarmente inaccessibile, (rileggete l'articolo sul numero 14) l'operazione presenta alcune difficoltà da superare.

Commodore 64: la piastra vista dall'alto.

Salvare direttamente tramite i normali comandi Basic (OPEN oppure SAVE usato nel modo per salvare il contenuto di una zona di memoria da voi scelta) non è possibile. Infatti la pagina grafica condivide la propria "casa" con il Sistema Operativo e quando cercate di leggere (con PEEK o direttamente tramite SAVE) il contenuto delle locazioni in questione è quest'ultimo che viene letto. Quindi il valore "peekato" non ha niente a che vedere con il contenuto dalla pagina grafica. L'operazione inversa, cioè "pokare", invece non presenta inconvenienti perché la POKE è sempre diretta alla R.A.M. (quindi alla pagina grafica) anche se "sopra" c'è una R.O.M. (il Sistema Operativo).

Ecco dunque la necessità di ricorrere ad una routine in linguaggio macchina che "tolga di mezzo" temporaneamente il Sistema Operativo e renda visibile in fase di lettura la sola pagina grafica. Ma se il Sistema Operativo viene disabilitato non è più possibile utilizzarne le comode routines dedicate alla comunicazione con le periferiche.

Risolto un problema ecco dunque che se ne presenta un altro. Le soluzioni sono due: o si riscrivono, con le opportune modifiche, le preziose routines di I/0 (mi vengono i brividi solo a pensarlo) oppure si trasferisce il contenuto della pagina grafica in una zona più accessibile e si riabilita il Sistema Operativo. Poiché non sono un masochista ho optato per la seconda soluzione.

Ma dove trasferire 8 K (circa) di memoria senza rischiare di invadere lo spazio riservato ai programmi in Basic? Naturalmente sotto la R.O.M. dell'interprete Basic.

Questa è mappata dalla locazione 40960 alla locazione 49151 e non interviene nelle operazioni di I/0. Può quindi venire disabilitata senza importanti conseguenze. Questo trasferimento della pagina ha permesso inoltre di realizzare l'istruzione GRMERGE che sicuramente vi farà comodo.

I comandi

Anche questi comandi, come i precedenti delle routines grafiche, vanno preceduti dalla solita freccetta (a meno che abbiate seguito le indicazioni pubblicate sul n. 16 per la sua eliminazione).

GRSAVE

La sintassi dei parametri è identica a quella del SAVE "normale". Esempio:

GRSAVE
GRSAVE "nome"
GRSAVE "nome", n (n=numero di periferica)

La pagina grafica viene salvata come un file programma. Sia questa che le altre istruzioni funzionano sia con il registratore che con il floppy disk driver.

GRLOAD

Sintassi dei parametri identica al LOAD normale. Esempio:

GRLOAD
GRLOAD "nome"
GRLOAD "nome", n
GRLOAD "nome", n, 1

L'indirizzo secondario non è necessario in quanto la pagina grafica caricata con questa istruzione viene SEMPRE messa negli ultimi 8 K di memoria.

Eventuali disegni presenti nella pagina al momento del GRLOAD vengono irrimediabilmente cancellati.

GRVERIFY

Sintassi dei parametri identica al VERIFY normale. Esempio:

GRVERIFY
GRVERIFY "nome"
GRVERIFY "nome", n

La verifica non viene effettuata comparando la registrazione con la pagina grafica "normale" ma con la sua immagine trasferita nelle locazioni 40960-48959. Immagine che rimane invariata fino al prossimo GRSAVE o GRLOAD.

GRMERGE

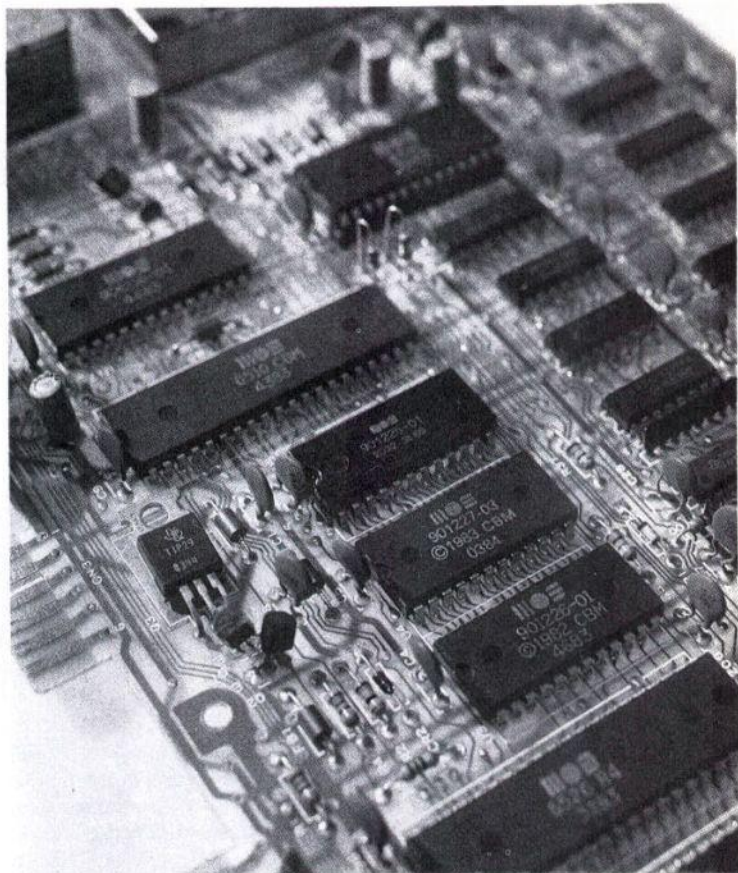
Sintassi dei parametri identica al LOAD. Esempio:
GRMERGE
GRMERGE "nome"
GRMERGE "nome", n
GRMERGE "nome", n, 1

Stesse considerazioni di GRLOAD riguardo l'indirizzo secondario. Questa istruzione consente di caricare dalla memoria di massa la pagina grafica e di sovrapporla a quella già presente nella memoria del calcolatore SENZA CANCELLARE quanto è già presente nella pagina grafica al momento del caricamento. Si ha cioè la possibilità di fondere le due pagine: quella residente e quella caricata.

Effetti collaterali

Come ho detto prima, il fatto di disabilitare il Basic non comporta gravi conseguenze ma qualche effetto si può notare se l'operazione di I/0 viene bruscamente interrotta.

Se usate il registratore e premete il tasto RUN/STOP quando compare la scritta PRESS PLAY... non succede nulla. Per fermare l'operazione dovete premere anche RESTORE. Se invece premete RUN/-STOP mentre lo schermo è "annebbiato" l'operazione viene interrotta ma invece di BREAK ERROR a volte viene visualizzato uno strano 3 ERROR, altre volte le scritte cambiano colore e appaiono in minuscolo.

Con il disco invece, se premete RUN/-STOP o se viene generato qualche errore di quelli che fanno lampeggiare la spia rossa si ottiene, generalmente, il cambio del colore delle scritte e un incomprensibile HRC ERROR. In ogni caso nulla di grave: basta premere RUN/STOP e RESTORE contemporaneamente e il computer torna a posto.

Commodore 64: particolare della piastra. Si può notare il microprocessore 6510.

Importante: nella frazione di secondo in cui è disabilitato il Sistema Operativo (con GRSAVE subito dopo che premete RETURN, con GRLOAD e GRMERGE alla fine del caricamento) non premete assolutamente il tasto RESTORE perché ottereste il blocco del computer.

SAVE e GRSAVE da programma

Come tutti saprete non è possibile effettuare un'operazione di salvataggio direttamente da programma. Basta però una semplice POKE e anche questo ostacolo viene superato.

Prima di ogni SAVE o GRSAVE inserito POKE 157,0 e subito dopo POKE 157,128. Esempio:
100 POKE 157,0:←GRSAVE "nome": POKE 157,128

Il contenuto di questa locazione infatti viene controllato dalla routine SAVE del Sistema Operativo: se contiene 0 significa che si è in modo diretto se invece contiene 128 vuole dire che sta girando un programma e il salvataggio non viene effettuato.
Non ho riscontrato inconvenienti nel manipolare questa locazione.

Gli altri comandi (GRLOAD, GRVERIFY, GRMERGE) possono essere usati all'interno dei programmi senza ricorre a "trucchi".

Il listato

Non vi resta ora che digitare il listato.

Se siete poco intraprendenti e non volete utilizzare il programma proposto nel n. 16 per aggiungere questi nuovi comandi alle routines grafiche, poco male: caricate que-

ste ultime e aggiungete in fondo al listato le linee dalla 5000 alla 9600.

Inserite poi la linea:

285 GOSUB 5000

e la linea:

9700 RETURN

Cambiate l'ultimo DATA della linea 2280 da 9 a 13 (indica il numero dei comandi presenti nella tabella delle parole). Di conseguenza dovete cambiare anche l'ultima somma di controllo della linea 225 che passa da 18956 a 18960 (se avete seguito i suggerimenti pubblicati per avere i comandi senza la freccetta la somma passa da 18963 a 18967).

Non vi resta a questo punto che dare il RUN e i quattro nuovi comandi saranno attivati.

Se invece siete ansiosi di utilizzare la sopracitata routine che abilita nuovi comandi non copiate le linee dalla 5600 alla 5900 e dalla 9100 a 9600 poiché i nomi e gli indirizzi dei comandi verranno inseriti nelle tabelle dal programma.

Gli indirizzi che dovrete inserire nelle "linee comando" sono specificati nelle linee dalla 20 alla 50. I nomi, come sapete, potete benissimo cambiarli.

Nota importante: le routines qui proposte funzionano esclusivamente se sono presenti in memoria le routines grafiche II poiché ne richiamano due subroutines.

Con ciò mi sembra di avere detto tutto perciò vi lascio alla vostra amata tastiera.

Danilo Toma

```
20 REM  ***  GRSAVE=51418     ***
30 REM  ***  GRMERGE=51468    ***
40 REM  ***  GRLOAD=51472     ***
50 REM  ***  GRVERIFY=51489   ***
60 :
5000 B=0:FORI=51380TO51559:READA:B=B+A:POKEI,A:NEXT
5100 IFB<>24684THENPRINT"ERRORE NEI DATA DELLE ROUTINES":END
5600 B=0:FORI=51164TO51183:READA:B=B+A:POKEI,A:NEXT
5700 IFB<>1440THENPRINT"ERRORE NEI DATA DELLE PAROLE":END
5800 B=0:FORI=50190TO50197:READA:B=B+A:POKEI,A:NEXT
5900 IFB<>1082THENPRINT"ERRORE NEI DATA DEGLI INDIRIZZI":END
6000 :
6001 REM    ****    ROUTINES    ****
7000 DATA 169,0,133,249,133,251,162,32,168,32
7100 DATA 94,197,32,88,201,177,249,36,251,145
7200 DATA 251,200,208,247,230,250,230,252,202,208
7300 DATA 240,32,109,197,32,96,201,96,104,104
7400 DATA 169,224,133,250,169,160,133,252,169,36
7500 DATA 141,197,200,32,180,200,32,212,225,32
7600 DATA 88,201,162,64,160,191,169,160,133,254
7700 DATA 169,0,133,253,169,253,32,216,255,32
7800 DATA 96,201,144,3,76,249,224,96,169,17
7900 DATA 208,2,169,36,141,197,200,169,224,133
8000 DATA 252,169,160,133,250,169,0,240,2,169
8100 DATA 1,133,10,104,104,32,212,225,32,88
8200 DATA 201,165,10,162,0,160,160,32,213,255
8300 DATA 32,96,201,176,205,165,10,240,10,162
8400 DATA 28,32,183,255,41,16,208,13,96,32
8500 DATA 180,200,32,183,255,41,191,240,245,162
8600 DATA 29,76,55,164,120,169,254,37,1,133
8700 DATA 1,96,169,1,5,1,133,1,88,96
9000 :
9100 REM    ****    PAROLE    ****
9200 DATA 3,71,82,148,3,71,82,147,3,71
9300 DATA 82,149,7,71,82,77,69,82,71,69
9400 :
9500 REM    ****    INDIRIZZI    ****
9600 DATA 218,200,16,201,33,201,12,201
READY.
```

IL CIRCUITO INTEGRATO 6561

Il mezzo più usato per conversare con un computer? Il video. Perché? È il più immediato. Osservando più computers accesi, ci si accorge però che il formato dello schermo varia da macchina a macchina.

In casa Commodore, ad esempio, lo schermo del C 64 (40 x 25) è addirittura doppio di quello del VIC 20 (22 x 23). Nel VIC 20, a gestire il formato dello schermo (e non solo quello) è preposto un integrato (Chip), detto 6561 VIC (Video Interface Chip) che, piccolo pettegolezzo di corridoio, è stato interamente progettato dalla COMMODORE.

I registri del 6561.

Il 6561 permette che il video faccia da intermediario tra utente e macchina.

Saranno quindi demandate a lui tutte le necessarie funzioni di controllo: formato dello schermo, colore, suono, controllo per la generazione dei caratteri e, (ma qui non ne parleremo) controllo per paddles e penna luminosa.

Per ogni funzione il 6561 usa una ben definita locazione di memoria RAM, ognuna delle quali prende il nome di registro. Ve ne sono 16 che partono dalla locazione 36864 alla 36879.

Registro n°1 (36864)

Il suo contenuto normale è 12 Print Peek (36864), ma, cambiandolo opportunamente, è possibile spostare lo schermo in orizzontale, a piacere. Digitate per esempio questo mini-programma:

100 PRINT "CLEAR" FOR I = ∅ TO 64 STEP 2: POKE 36864, I: FOR T = 1 TO 50: NEXT: NEXT.

Il ciclo principale incrementa con passo 2 (ad ogni passo 2 lo schermo si sposta di un carattere), partendo da ∅, il contenuto della 36864. Il secondo ciclo è ovviamente di ritardo.

Il risultato è uno schermo che si muove da sinistra a destra fino a scomparire del tutto. Per riavere lo schermo battete adesso, come pure per tutti gli altri mini-programmi, che saranno tra breve proposti RUN STOP e RESTORE. Si noti che il ciclo va da ∅ a 64, in quanto il numero massimo di colonne, ottenibile da lato a lato, è 26. Il ciclo si ripeterebbe allo stesso modo anche se si partisse da 128 anziché da ∅.

La più evidente applicazione pratica è quella di centrare orizzontalmente lo schermo, dopo aver agito opportunamente sui registri che ne controllano il formato.

Registro n°2 (36865)

Il suo contenuto normale è 38, e controlla la centratura verticale. Digitate questa linea:

100 PRINT "CLEAR": FOR I = ∅ TO 166 STEP 4: POKE 36865, I: FOR T = 1 TO 50: NEXT: NEXT.

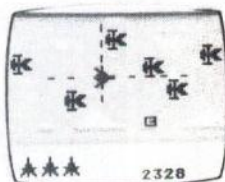
La differenza fra i due registri è che quest'ultimo si muove a passo 4 anziché 2 e quindi il ciclo, anche se impostato da ∅ a 255, si ripete una volta sola. È ovvio che il movimento dello schermo può essere invertito sia in modo verticale che orizzontale.

Registro n°3 (36866)

Il suo contenuto normale è 150. Con questa locazione di memoria si può ottenere un numero variabile di colonne da 1 a 26.

Prima di tutto, vá considerato che non è possibile avere un formato di schermo che abbia un numero di bytes superiore a 512, a meno di usare opportune modifiche, come illustrate nell'articolo *Scrittura a tutto schermo* di Marco Navalesi, apparso su C.C.C. n°11 (tenere presente solo le linee da 120 a 180).

Quindi, se si vogliono avere più colonne del normale, ed usando solo i 512 bytes messi a disposizione dal sistema, bisognerà prima diminuire il numero delle righe. Il discorso vale anche per l'inverso: più righe meno colonne.

Il metodo da usare è quello di aggiungere a 128 tante unità quante sono le colonne

che si vogliono avere sul video (infatti 128 + 22 colonne = 150). Come esempio battete questo programma:

100 PRINT "CLEAR": POKE 36864,8: POKE 36867,166: FOR I = 129 TO 154: POKE 36866,I: FOR T = 1 TO 100: NEXT: NEXT.

Per prima cosa viene spostato tutto lo schermo a sinistra. Poi, per avere una corretta esecuzione del programma, viene diminuito il numero di righe con il registro n°4, (che vedremo tra breve) quindi il solito ciclo principale che parte da 129 (128 di base + la prima colonna) fino a 154 (come si è detto le colonne sono al massimo 26).

Registro n°4 (36867)

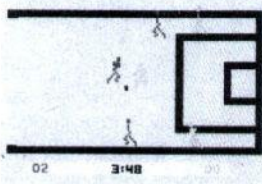
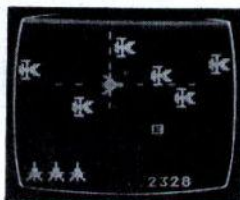
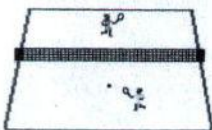
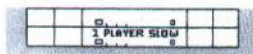
Il suo normale contenuto è 174. Si è visto nell'esempio precedente il modo per definire il numero delle righe, che è simile a quello ora usato. Si deve aggiungere a 128 il numero di righe moltiplicato per 2 (infatti 128 + 23 righe x 2 = 174). Battete questo programma:

100 PRINT "CLEAR": POKE 36865,20: POKE 36866,143: FOR I = 130 TO 192 STEP 2: POKE 36867,I: FOR T = 1 TO 100.

100 NEXT: NEXT.

Prima lo schermo viene spostato in alto, poi viene ridotto il numero di colonne, e quindi il solito ciclo che parte da 130 (128 di base + la prima riga x 2) fino a 192 (128 + 32 x 2), dove 32 è il massimo numero ottenibile di righe. È necessario tenere presente che il contenuto del registro n°4 deve essere sempre pari, in quanto il bit N.∅ controlla quale tipo di matrice dei caratteri deve essere usata. Infatti quando il bit ∅ è settato (= 1), la matrice sarà di 16 x 8, contro la normale 8 x 8, che si ottiene quando il bit ∅ è pulito (=∅).

A titolo di esempio, togliete dall'ultimo mini-programma l'istruzione STEP 2; il risultato è abbastanza suggestivo: lo schermo si muove verso il basso con movimenti sussultorii. Questo avviene perché, quando il contenuto della 36867 è dispari, l'altezza del carattere è doppia del normale. Per lavorare in Hi-Res, è ovviamente di grande utilità agire su questo registro.

Vogliamo divertirci un po'? Bene, allora mantenete in memoria il programma, ma, prima di dare il RUN, cambiate il colore dello schermo con POKE 36879,9 e togliete lo STEP 2. Visto? Lo schermo si riempie di picche e poi torna nero, ripetendosi per tutto il ciclo.

Registro n°6 (36869)

Il suo contenuto normale è 240. I suoi primi 4 bits (da ∅ a 3) determinano il punto di inizio per la lettura del generatore di caratteri. Quando non viene alterato il suo contenuto, viene puntato il generatore standard, che parte dalla 32768 in avanti per 4K ROM. Tuttavia è possibile cambiare i caratteri del VIC con altri di diversa fattura; agendo di conseguenza su questa locazione, si potrà puntare al nuovo generatore.

Avendo il VIC solo 3,5K RAM liberi, non è possibile cambiare tutti i caratteri del generatore standard. Per esigenze di spazio è addirittura consigliabile non superare i 64 caratteri che costituiscono un set grafico.

A titolo di esempio battete questo programma che non genera caratteri nuovi, ma semplicemente serve a capire come funziona la 36869:

100 PRINT "CLEAR": POKE 36879,9: FOR I = 7168 TO 7679: POKE I, PEEK (25600 ± I): NEXT: POKE 36869,255

110 FOR I = ∅ TO 63: POKE 7680 + I,I : NEXT: PRINT SPC (255).

Per prima cosa è utile cambiare il colore dello schermo. In una normale istruzione di POKE, il carattere viene visualizzato in bianco, rendendolo quindi invisibile allo schermo base del VIC. Poi vengono letti i contenuti del primo blocco di 512 bytes del generatore standard (da 32768 a 33279) e trasferiti nell'ultima parte di memoria libera (da 7168 a 7679). Quindi viene predisposto il registro n° 6 in modo che siano letti i caratteri del nuovo generatore, cioè dalla 7168 in poi. L'ultimo ciclo, infine, visualizza i primi 64 caratteri, che diciamo ridefiniti, anche se non è ovviamente vero.

Ora, però, modificate il ciclo FOR.-..NEXT della linea 110 sostituendo 63 con 127. Il risultato è differente se cambiamo il contenuto della 36869 con 240 (modo normale) e con 255. Con il primo vengono letti regolarmente, mentre con il secondo solo i primi 64 sono quelli che ci aspettiamo. Infatti ricercando gli altri 64 caratteri, il S.O. va a leggere i contenuti delle locazioni da 7680 in poi per 512 bytes (la memoria di schermo).

Per potere leggere anche il secondo set di caratteri, bisognerà modificare il programma come segue:

100 PRINT "CLEAR": POKE 36879,9: FOR I =6144 TO 7167: POKE I, PEEK (26624 ± I): NEXT: POKE 36869,254

110 FOR I = ∅ TO 127: POKE 7680 ± I,I: NEXT.

Come si può notare da queste linee, il nuovo generatore dovrà essere "puntato" alla locazione 6144, mentre la lettura di

quello standard sarà sempre da 32768 (26624 + 6144) in poi. È variato anche il contenuto della 36869, proprio perché questo registro punta a locazioni ben determinate. La tabellina pubblicata spiega come operare per ottenere il puntamento (tenere presente che i bits da 4 a 7 della 36869 devono essere tutti settati).

(si parte dallo ∅) del secondo set in REVERSE.

Registri n°7 (36870) e n°8 (36871)

Sono i due registri che controllano rispettivamente la posizione orizzontale (X) e verticale (Y) della penna luminosa.

Bit	3	2	1	∅			Start per generatore caratteri
	∅	∅	∅	∅	=	240	32768 (generatore standard RAM)
	1	1	1	1	=	255	7168 (generatore RAM)
	1	1	1	∅	=	254	6144 (generatore RAM)
	1	1	∅	1	=	253	5120 (generatore RAM)
	1	1	∅	∅	=	252	4096 (generatore RAM)

Se in un programma vengono utilizzati caratteri standard in REVERSE e caratteri nuovi (è solo un esempio), bisognerà cambiare tattica. Leggere i contenuti della 32768 in poi non serve a niente, a meno che non si punti alla 5120, allocando tutto il primo blocco di 2K del generatore standard. Poi però resterebbe 1K solo libero. Si può invece avere lo stesso risultato utilizzando solo 512 bytes. Provate a digitare questo programmino:

100 PRINT "CLEAR": FOR I = 7168 TO 7679: POKE I, PEEK (26624 + I): NEXT: POKE 36869,255

110 PRINT "BLK HOME ABCDEFGHIJKLMNOPQRSTUV".

Per prima cosa vengono letti i contenuti da 33792 (e non più da 32768) in poi. Da questa locazione parte il generatore standard dei caratteri in REVERSE. Questi contenuti vengono allocati dalla 7168 in poi. La stringa della linea 110 dimostra che siamo realmente nella zona REVERSE, senza avere aggiunto il tasto RVS ON. Il discorso vale anche se, invece di volere il primo set in REVERSE, si vuole il secondo. È sufficiente sostituire a 26624 della linea 100 il valore 27136 (27136 + 7168 = 34304 start per secondo set in REVERSE).

Attenzione, però, la stringa della linea 110 deve essere mantenuta tale e quale. Il sistema legge il carattere "A" (codice ASCII) che, per come è impostato il nuovo generatore, corrisponde al carattere n° 1

Registri n°9 (36872) e n°10 (36873)

Questi registri controllano invece le PADDLES.

Registri dal n°11 al n°15 (da 36874 a 36878)

Sono registri noti a tutti, che controllano il rumore bianco ed il volume del VIC. La frequenza degli oscillatori viene modificata con valori da 128 a 255, mentre per il volume con valori da ∅ a 15.

Registro n°16 (36879)

Questo famigerato registro è comune in tutti, o quasi, i programmi. È inutile soffermarci sulla sua funzione. Nel modo normale determina il colore di fondo e di margine dello schermo.

Ancora sul formato

Come abbiamo visto in precedenza, i primi 4 registri controllano il formato dello schermo. Supponiamo ora di avere uno schermo di 15 x 15. Va considerato che, per il sistema, il numero delle righe ha poca importanza, mentre il numero delle colonne può creare qualche problema. (Il sistema operativo deve sapere quando andare a capo). Comunque, predisponiamo lo schermo con questa linea:

100 PRINT "CLEAR": POKE 36864,18: POKE 36865,52: POKE 36866,128 + 5: POKE 36867,128 + 15 x 2.

Oramai non ci sono più segreti. Il lavoro diventa problematico se in questo schermo si vuole scrivere una oppure più stringhe. Come si è detto, il S.O. tiene sempre conto che le colonne sono 22. Il controllo è determinato dalla locazione 213 che, essendo in pagina zero, è in pagina zero, è in memoria RAM.

Sostituire al valore 22, contenuto normale, un numero pari alle colonne volute, non serve a niente in quanto, non appena viene attuata la modifica, il S.O. restituisce 22 alla locazione 213. Bisogna trovare un altro sistema. Il più dinamico è quello di usare stringhe lunghe quanto il numero delle colonne e separarle tra loro da un punto e virgola.

Ad esempio, continuando con il programmino di cui sopra, aggiungete questa linea:

110 PRINT "HOME ABCDEFGHIJKLMNO": PRINT "ABCDEFGHJKLMNO".

Il risultato, se volevamo le lettere ben incolonnate, è piuttosto deludente. Ma, se al posto dei due punti, mettiamo un punto e virgola e togliamo il secondo PRINT, otteniamo il risultato richiesto.

Un altro modo è quello di unire le due stringhe in una, ma alla fine, se si devono scrivere altre stringhe, l'uso del punto e virgola è inevitabile. Il modo più veloce è sempre quello di usare l'istruzione TAB (X). Modificate la linea 110 con:

110 PRINT "HOME" TAB (∅) "ABCDEFGHIJKLMNO": PRINT TAB (23) "ABCDEFGHIJKLMNO".

È evidente che prima si devono fare alcuni calcoli, ma l'esecuzione è più veloce, con un buon risparmio di memoria, proporzionale al numero delle stringhe. Per i calcoli si tenga sempre presente che il VIC considera il 22 come standard fisso.

Il Breakout è un'applicazione di quanto sopra detto; è tutto in linguaggio macchina, quindi lunga e noiosa la sua spiegazione. Se avrete la pazienza di digitare i due programmi (dovranno essere registrati uno dopo l'altro), potrete vedere uno schermo modificato. Addirittura viene usato uno schermo rovesciato, verticale anziché orizzontale.

conseguenze imprevedibili e non sempre coincidenti con l'inchiodamento del sistema.

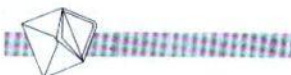
Il rimedio, in questo caso, consiste nello smontare l'apparecchio, rintracciare le saldature "sospette" e ripassarle col saldatore. Sconsigliamo caldamento tale intervento che potrebbe provocare, se affidato a persone poco esperte di saldature con circuiti integrati, corti circuiti con conseguenze irreparabili. Affidarsi a personale più che esperto.

• Il cursore "passeggia" per lo schermo senza che sia comandato da tastiera o da joystick. Premendo alcuni tasti appaiono simboli diversi di quelli raffigurati sui tasti stessi.

Tale inconveniente compare spesso utilizzando, o semplicemente connettendo, alcuni joystyck che contengono al loro interno un circuito integrato che consentirebbe, stando alle dichiarazioni dei costruttori, velocità maggiori con videogames. Purtroppo abbiamo notato che tali accessori generano effetti collaterali indesiderati. Come rimedio consigliamo, se presenti nelle apposite prese, di staccare i joy in questione: se il difetto scompare é evidente che tutto dipende dall'accessorio.

• L'unità a dischi (e/o il registratore) carica male programmi che in certe occasioni carica invece al primo colpo. Il difetto é dovuto in molti casi al fatto che la memoria di massa (registratore o drive) oppure il solo cavetto di collegamento, si trovano troppo vicini al trasformatore di alimentazione del computer oppure al televisore.

In questi casi il rimedio consiste nel tenere lontani apparecchi e cavi dal TV oppure nel realizzare schermi protettivi ricorrendo agli economici, quanto efficienti, fogli di alluminio del tipo di quelli utilizzati normalmente nella conservazione dei cibi.

Se i rimedi suggeriti non apportano migliorie vuol dire, ahinoi!, che é necessario inviare il computer a centri specializzati per riparazioni.

Computer con la febbre

☐ **Il mio 64 si scalda molto. Quale é il limite di sicurezza?**

● Purtroppo la Commodore non ha diffuso le temperature ottimali di lavoro dei suoi apparecchi. Penso, però, che ti riferisci particolarmente al trasformatore di alimentazione dato che il calcolatore, a toccarlo da ogni parte, sembra praticamente "freddo".

Per curiosità ho provato a toccare gli apparecchi che abbiamo in dotazione, e quello che scalda di più é, come ho già detto, il trasformatore di alimentazione. Questo deve fornire un tepore intenso.

Se la risposta non ti soddisfa tieni presente che una temperatura eccessiva é dovuta a surriscaldamento da cortocircuito. Se, però, fosse stato questo il tuo caso il fusibile di protezione si sarebbe bruciato: é fatto apposta!!.

Basic 4.0

☐ **Che cosa é il Basic 4.0?**

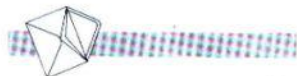
● È una utility che, caricata da disco, aggiunge al Commodore 64 numerosi comandi aggiuntivi che hanno la stessa sintassi del linguaggio usato nei computer Commodore della serie 4000 e 8000. In particolare facilita l'uso di due drive, rende semplice l'utilizzo di file relative, legge la directory senza cancellare il programma in memoria eccetera.

Protezione dei programmi

☐ **In che modo posso sproteggere i programmi posti in vendita (per effettuare duplicazioni)?**

● Siamo a conoscenza dei vari sistemi di sprotezione ma non li pubblichiamo per motivi di salute: dicono infatti che le patrie galere siano umide.
Scherzi a parte, ripetiamo ciò che abbiamo detto più volte:

• Un discorso sulle protezioni (e non sulle sprotezioni) é stato iniziato sulla rivista Micro & Personal Computer.

• Molti sprotettori, che pongono in vendita a poche lire programmi costosissimi, incominciano a esser raggiunti da denuncie da parte di software house che si ritengono danneggiate.

Caricamenti irregolari col 1541

☐ **A volte il caricamento di un programma registrato precedentemente su disco richiede un tempo enorme rispetto al consueto e produce un lampeggio del LED. Come mai?**

● Anche i drive, come i registratori, hanno problemi di allineamento delle testine di lettura/scrittura. Sono fortunatamente, casi infrequenti. Prova a tener lontano il tuo drive dal TV oppure verifica che lo sportellino si chiuda correttamente. Se, infatti, il difetto lamentato scompare tenendolo premuto durante la fase di caricamento, é probabile che la molla di chiusura si sia allentata.

Ti assicuri, comunque, di verificare i programma subito dopo averlo registrato? Spesso il difetto lamentato si verifica con file non allocati correttamente su disco a causa di una cattiva registrazione.

Programmi troppo protetti

☐ Col Commodore 64 carico normalmente programmi non protetti e con gran difficoltà quelli protetti. Come mai?

● Purtroppo alcune protezioni di cui sono dotati certi programmi hanno effetti collaterali indesiderati come, appunto, anomalie di caricamento.

Se sei certo che il difetto non é dovuto ad un cattivo allineamento delle testine, alla vicinanza del trasformatore o del TV, purtroppo non c'é nulla da fare.

Conversione da L.M. in Basic

☐ È possibile convertire automaticamente un programma scritto in Linguaggio Macchina nel linguaggio Basic?

● Assolutamente no. Un listato L.M. é infatti composto da una moltitudine di istruzioni elementari tra cui molte istruzioni di salto a subroutine in cui, spesso, vi sono altri salti ad altre subroutine.

Un programma che converta automaticamente dovrebbe tener conto di troppe situazioni particolari occupando, di conseguenza, considerevole spazio in memoria.

Solo ricorrendo a notevole pazienza é possibile, e non sempre, studiare il disassemblato del programma e cercare di ottenere lo stesso "effetto" di segmenti di programma ricorrendo ad istruzioni Basic.

Compilatori

☐ Che cosa é un compilatore?

● Un compilatore é un particolare programma che esamina un listato Basic e lo trasforma automaticamente in Linguaggio Macchina. L'effetto che ne deriva é un aumento della velocità di esecuzione nei casi in cui il Basic presentava notevoli limiti.

In commercio esistono due compilatori: il Petspeed e l'Austrospeed, entrambi

funzionanti solo col Commodore 64 usato insieme col drive 1541.

Il loro funzionamento é relativamente semplice: si registra il programma in Basic da compilare sul dischetto che contiene il compilatore ed in seguito, fatto partire quest'ultimo, si seguono le istruzioni che compaiono a mano a mano sul video. Alla fine dell'intera procedura (che dura, a seconda dei casi, un tempo non minore di un quarto d'ora), sullo stesso dischetto verrà creato un File-Programma che rappresenta il risultato dell'elaborazione.

Questo file può essere caricato come un qualsiasi programma Basic ricorrendo all'istruzione LOAD e può essere trasportato anche su cassetta. Per il suo funzionamento, infatti, non é più necessario il programma compilatore, utilizzato, del resto, nella sola fase di creazione.

I programmi compilati si individuano facilmente perché, digitando LIST, appare semplicemente SYS(XXX): NOME in cui NOME é il nome commerciale del programma che ha generato il file.

Istruzioni SYS

☐ Posseggo programmi molto lunghi da caricare, ma costituiti da una sola istruzione: SYS(N) in cui N é un numero che non é sempre lo stesso. Che cosa fa esattamente SYS, dato che lo trovo in molti programmi Basic?

● Il comando SYS comunica al computer che deve "abbandonare" il linguaggio Basic ed operare direttamente in Linguaggio Macchina tenendo conto che la prima istruzione da eseguire é contenuta nel byte il cui indirizzo é indicato tra parentesi.

Il Linguaggio Macchina é un linguaggio difficilissimo da interpretare perché lavora interagendo in modo diretto col "cervello" dell'elaboratore.

Se, dunque, un programma richiede molto tempo per il caricamento ed é formato da una sola istruzione SYS, il listato vero e proprio é scritto in L.M. e, di conseguenza impossibile da listare per il semplice motivo che... non c'é (in Basic).

Un programma particolare, detto Disassembler, facilita l'interpretazione ma richiede, come puoi intuire, la conoscenza approfondita del linguaggio e del sistema operativo del calcolatore.

Autostart

☐ È possibile far partire un programma non appena viene caricato da cassetta?

● Premento il tasto Shift insieme con Run/Stop si ottiene l'effetto che cerchi.

Se invece desideri che il tuo programma parta anche senza questo accorgimento, ti consiglio di leggere la rivista Micro & Personal Computer su cui viene, da un po' di tempo, trattato il tema della protezione dei programmi.

Turbo tape & dintorni

☐ Programmi caricati col turbo tape richiedono lo spegnimento della macchina per interromperli e caricare altri programmi. C'é un sistema che eviti lo spegnimento e il ricaricamento del Turbo?

● Il problema non é del turbo tape ma del sistema di protezione di cui son dotati i programmi commerciali.

All'inizio di tali listati, infatti, sono presenti particolari istruzioni che inibi-

scono il funzionamento dei tasti Run/ Stop e Restore in modo da evitare interruzioni e possibilità di copie non autorizzate.

L'impossibilità di interrompere un programma é pratica molto diffusa anche in programmi normali, non "turbizzati".

Memoria a volontà

□ **Che si intende col termine: "fissare il Top di memoria"?**

● Supponiamo, tanto per semplificare, che il tuo computer abbia, appena acceso, una quantità di memoria RAM pari a 10000 byte.

Volendo caricare un programma lungo 3000 byte sembrerebbe che, dopo l'operazione, restino a disposizione 7000 byte (10000-3000). Non dobbiamo, però, dimenticare che quando il programma vien fatto girare, nella maggior parte dei casi vengono dichiarate matrici (DIM), variabili intere (come B=10), variabili in virgola mobile (come A=56.5464). Ogni variabile sottrae memoria RAM pari a sette byte mentre per le matrici si seguono calcoli diversi.

Per renderti conto di ciò che segui quanto ti dico:

- Accendi il computer e prendi nota della quantità di RAM presente.
- Carica un qualsiasi programma in Basic e prendi nota della quantità di RAM, rimasta dopo tale operazione, con: PRINT FRE (0).
- Fai partire il programma (RUN) e, ad un certo punto, interrompilo con Run/ Stop e digita nuovamente: PRINT FRE(0).

Ti renderai conto che la quantità é minore di quelle precedenti.

Sembrerebbe, a questo punto, che il numero di byte liberi possa esser calcolato tenendo conto dei byte occupati dal programma, da quelli occupati dalle matrici e da quelli occupati dalle variabili.

Purtroppo non é così perché, per motivi che non sto qui a dirti, tutte le volte che vengono elaborate variabili stringa, il calcolatore utilizza la parte di memoria RAM "eccedente".

Ciò vuol dire che se trascrivi alcune informazioni (come una routine in L.M. da richiamare mediante SYS) in una zona che ritieni libera, il computer, a mano a mano che elabora le stringhe, può trascriverle in quelle locazioni alterandole in modo irreparabile. Per fortuna c'é un modo per riservarsi una zona RAM e

renderla inattaccabile dal Sistema Operativo del computer: consiste nel fissare il tetto massimo della memoria RAM.

Il calcolatore, infatti, elabora le stringhe tenendo conto che la memoria presente é tutta quella disponibile al momento dell'accensione dell'apparecchio. È però possibile "ingannare" il computer facendogli credere che la memoria RAM é più esigua alterando due puntatori che prendono, appunto, il nome di Top Of Memory. Questi sono rappresentati dal contenuto delle locazioni 55 e 56 modificabili a piacimento mediante istruzioni POKE. Se digiti, non appena accendi l'apparecchio:

PRINT PEEK(56): PRINT FRE(0): POKE 56,20: CLR: PRINT FRE (0) otterrai due risposte completamente diverse. Tranquillizzati, però. La memoria RAM é sempre lì, al suo posto, solo che il Sistema Operativo non la "vede" e, di conseguenza, non la altererà nelle sue elaborazioni. È ovvio che puoi inserirvi ciò che vuoi ma tieni conto che i programmi basic che puoi ora caricare son più brevi.

Benché superfluo, ricordo che 55 contiene il byte basso e 56 il byte alto dell'indirizzo dell'ultimo byte libero a disposizione del S.O. Per esempio con:

POKE 55,30: POKE 56,10: CLR

la memoria libera, a tua completa disposizione, é quella che "parte" dal byte il cui indirizzo é dato dal calcolo:

30+10*256=2590

Un ultimo consiglio: tali alterazioni effettuate prima di caricare un programma oppure come prima istruzione del programma stesso. Ti consiglio, comunque, di rileggere il N.10 di C.C.C. (Un po' d'ordine tra i bit).

Duplicatore di cassette

☐ **Alcuni lettori hanno avuto difficoltà a procurarsi il duplicatore di cassette presentato sul N. 15 di C.C.C. Altri, invece, che hanno chiesto per posta l'accessorio, si son visti recapitare un altro apparecchio.**

• Il duplicatore di cassette provato da Commodore Computer Club è commercializzato dalla Ditta R.C.P. Elettronica Srl Via Don Pasquino Borghi, 13 cap 42017 Novellara (Reggio E.) Tel. 0522/661471.

La stessa Ditta figurava, come inserzionista, a pagina 6 dello stesso fascicolo di C.C.C. proponendo, però l'interfaccia per adattare ben due registratori qualsiasi ai computer Vic 20 e Commodore 64.

Dalla fotografia della pubblicità, dal tagliando che era necessario compilare e dai disegni che corredavano l'articolo incriminato, il lettore avrebbe dovuto capire che si trattava di due accessori COMPLETAMENTE diversi.

A coloro che hanno telefonato per conoscere il nominativo della Ditta che vendeva l'accessorio abbiamo specificato quanto detto in queste note.

L'unica colpa (se tale si può definire) della Ditta R.C.P., consiste nel fatto di aver individuato nell'interfaccia, più che nel duplicatore di cassette, l'oggetto più idoneo da presentare ai lettori della rivista.

Videogames

☐ **Esiste un catalogo generale di giochi per il Commodore 64?**

● No. Vi sono infatti molte ditte, in concorrenza tra loro, ognuna delle quali mette però a disposizione del pubblico un catalogo completo. Rivolgiti a negozi specializzati in personal computer: troverai di certo ciò che cerchi.

Alessandro de Simone

COMPUTER

PRODOTTO	PREZZO (IVA ESCL.)
VIC 20	
HOME COMPUTER Memoria base 5 Kbytes, 255 combinazioni di colori, 3 voci, BASIC residente, si collega direttamente a qualsiasi televisore domestico.	199.000
COMMODORE 16	
HOME COMPUTER Memoria a 16 Kbytes RAM standard, 12 Kbytes RAM accessibili per programmi in BASIC. BASIC 3.5 residente. 121 colori (15 di base con 8 gradazioni + il nero). Collegabile a un comune televisore.	245.000
COMMODORE 64	
CPU 64K RAM Computer con alta risoluzione grafica, 256 combinazioni di colori, sintetizzatori di suono. Possibilità di creare eccezionali figure tridimensionali dinamiche (sprites). Tastiera grafica. Dimensione dello schermo 40 colonne per 25 righe. Compatibile con tutte le periferiche Commodore. Collegabile a un comune televisore.	625.000
C 64 EXECUTIVE Il solo e fantastico computer portatile della seconda generazione. Monitor a colori di 5" ad alta risoluzione grafica e Floppy Disk Drive da 170 Kbytes incorporati! Tastiera grafica separabile, suono professionale e completa compatibilità con i programmi e le periferiche del Commodore 64 ne fanno un computer dai mille usi. Può inoltre utilizzare tutte le cartucce C 64 e può essere collegato al monitor 1702.	2.350.000
COMMODORE PLUS 4	
PERSONAL COMPUTER Memoria 64 Kbytes RAM, 60 bytesK RAM accessibili per programmi in BASIC. BASIC 3.5 residente. 121 colori (16 di base con 8 gradazioni). 4 programmi software residenti su ROM. Collegabile a un comune televisore.	975.000
COMMODORE 8000	
PERSONAL COMPUTER CPU 128 Kbytes RAM, 24Kbytes ROM, Basic 4.0 residente, video orientabile e basculante 80 colonne per 25 righe, tastiera commerciale separata. Software compatibile con CBM 8032 e CBM 8096.	1.995.000
COMMODORE 700	
PERSONAL COMPUTER CPU 128 Kbytes RAM espandibili internamente a 256 Kbytes ed esternamente a 960K. Video orientabile e basculante, 80 colonne per 25 righe. Compatibile con tutte le periferiche Commodore delle serie professionale.	2.850.000
PERSONAL COMPUTER CPU 256 Kbytes RAM espandibili esternamente a 960 Kbytes. Video orientabile basculante, 80 colonne per 25 righe. Compatibile con tutte le periferiche Commodore delle serie professionale.	3.250.000

PERIFERICHE

PRODOTTO	COMPUTER COLLEGABILI	PREZZO (IVA ESCL.)
MONITOR		
MONITOR MONOCROMATICO A fosfori verdi 12"	VIC 20, C 16, C 64, SX 64, PLUS 4	285.000
MONITOR A COLORI Ad alta risoluzione, 14", con audio.	VIC 20, C 16, C 64, SX 64, PLUS 4	690.000
REGISTRATORI		
REGISTRATORE DEDICATO Per memorizzare facilmente programmi e dati su normali cassette magnetiche.	VIC 20, C 64,	120.000
REGISTRATORE DEDICATO Per memorizzare facilmente programmi e dati su normali cassette magnetiche.	C 16, PLUS 4	120.000
MEMORIE DI MASSA		
FLOPPY DISK DRIVE Unità di memoria di massa, drive singolo, capacità 170 Kbytes in linea.	VIC 20, C 16, C 64, SX 64, PLUS 4	630.000
FLOPPY DISK DRIVE Unità di memoria di massa, drive doppio, capacità 1 Mbytes in linea.	C 8032SK, C 8096, C 8296, C 710, C 720, C 610, C 620	2.375.000
FLOPPY DISK DRIVE Unità di memoria di massa, drive singolo, capacità 1 Mbytes in linea.	C 8032SK, C 8096, C 8296, C 710, C 720, C 610, C 620	1.245.000

INVIARE TUTTA LA PAGINA ANCHE SE SI UTILIZZA UNA SOLA SCHEDA

Nome .... Cognome ....

Via .... N° .... CAP. .... Città ....

Telefono .... Orario ....

Registrate il mio abbonamento annuale a Commodore Computer Club.

☐ Ho versato oggi stesso il canone di L. 22.000 a mezzo c/c postale n° 31532203 intestato a:
Commodore Computer Club - V.le Famagosta, 75 - 20142 Milano

☐ Ho inviato oggi stesso assegno bancario n..
per l'importo di L. 22.000 intestato a Commodore Computer Club.

Si prega di scrivere il proprio nome e l'indirizzo completo in modo chiaro e leggibile. Inviare la fotocopia del bollettino di c/c postale.

Consideriamo che i numeri 1 e 2 sono esauriti, vogliate inviarmi i numeri arretrati .... al prezzo di L. 5.000 cadauno per richieste fino a 4 numeri, o di L. 4.000 cadauno per richieste oltre i 4 numeri arretrati, e perciò per un totale di L............... Sono a conoscenza che i fascioli suddetti non saranno inviati in contrassegno e, pertanto, ho provveduto oggi stesso a versare il canone di L................ a mezzo c/c postale n. 31532203 intestato a:
Commodore Computer Club - V.le Famagosta, 75 - 20142 Milano

STATISTICA

Non posseggo un computer ☐

Posseggo un C64	si ☐	... no ☐
Posseggo un VIC 20	si ☐	... no ☐
Posseggo un Commodore Plus 14	si ☐	... no ☐
Posseggo un Commodore Plus 16	si ☐	... no ☐
Posseggo un registratore dedicato	si ☐	... no ☐
Posseggo un drive 1541	si ☐	... no ☐
Posseggo una stampante	si ☐	... no ☐
Posseggo un monitor	si ☐	... no ☐

COLLABORAZIONE

A titolo di prova vi invio un articolo e la cassetta.........................disco...................................
col programma che intendo proporre per la pubblicazione di cui garantisco l'originalità.

DOMANDA/RISPOSTA ....

....

....

....

....

....

RICHIESTA ARGOMENTI

Mi farebbe piacere che Commodore Computer Club parlasse più spesso dei seguenti argomenti:

1/ ....

2/ ....

3/ ....

4/ ....

GIUDIZIO SUI PROGRAMMI DI QUESTO NUMERO

Ho assegnato un voto da 0 a 10 ai programmi che indico di seguito:

A/ .... Voto ....

B/ .... Voto ....

C/ .... Voto ....

D/ .... Voto ....

PICCOLI ANNUNCI ....

....

....

....

....

....

CERCO/OFFRO CONSULENZA ....

....

....

....

....

....

INVIARE IN BUSTA CHIUSA E AFFRANCANDO SECONDO LE TARIFFE VIGENTI A:

COMMODORE COMPUTER CLUB

V.le Famagosta, 75
20142 Milano

INVIARE TUTTA LA PAGINA ANCHE SE SI UTILIZZA UNA SOLA SCHEDA

Nome ______________ **Cognome** ______________

Via ______________ **N°** ______ **CAP.** ______ **Città** ______________

Telefono ______________ **Orario** ______________

Computer School

L'esperienza insegna.

La richiesta di corsi d'informatica cresce ogni giorno di più. Ma non basta conoscere bene il computer per saperne insegnare l'uso ad un pubblico tanto ansioso d'apprendere quanto privo di qualsiasi nozione di base in materia. Nè la sola esperienza didattica è sufficiente per entrare in questo settore. Così, Incalzato dalla domanda, anche tu che non vuoi rispondere con un insegnamento insufficiente o improvvisato, probabilmente sei alla ricerca d'una metologia provata e sicura, di una manualistica coerente e completa, di sussidi audiovisivi e schemi di lavoro. Computer School R possiede una solida, tangibile e collaudata esperienza d'insegnamento dell'informatica e ti dà tutto questo, insieme al know-how ed al supporto necessario perchè anche tu possa entrare con successo in questo promettente mercato. Inoltre, consentendoti di presentarti agli utenti potenziali con il suo marchio e la sua insegna, ti offre un ulteriore vantaggio: beneficiare d'una possente azione pubblicitaria sulle più diffuse e prestigiose pubblicazioni del settore. Se perciò vuoi essere la prima Computer School della tua città non esitare a contattarci.

Franchising per insegnare.

20090 Trezzano S/N (MI) - V.le C. Colombo, 49 - Tel. (02) 4454352/4459252